平成３年６月２日第３種郵便物認可　平成７年７月１日発行(毎月１回１日発行)　第７巻　第７号　通巻第62号

TRY COMPUTING
トラ技コンピュータ

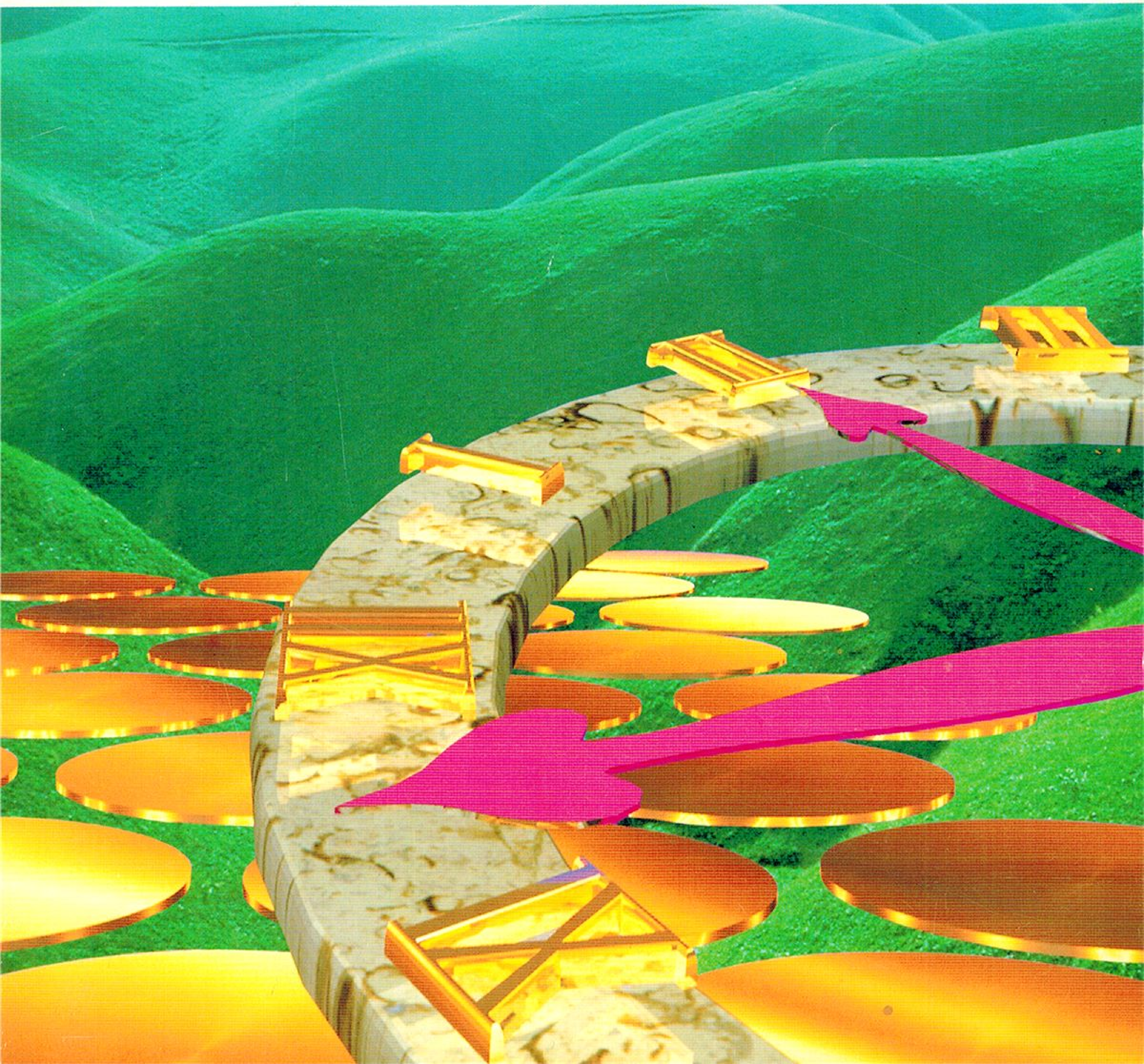
PDPからIBM PC，PC98へと受け継がれる技術を理解しよう

特集 知っておきたいミニコン/パソコン変遷史

1995 7

特設記事　プログラミングの世界を広げるLISPをマスターしよう

解　説　Paradox for Windows ObJect PAL簡単入門!



NEC



世界を先取る。

最新Pentium™プロセッサ&PCIバス搭載。
ビジネスの主力98MATE Xシリーズが、次世代OS Windows®95時代に向けてさらにパワーアップ。いま新たに生まれ変わった。

■先進Pentium™プロセッサ&PCIバス搭載。次世代先取りの高機能。

次世代OS Windows®95に向けて開発された新98MATE Xシリーズ。Windows®3.1環境はもちろん日本語版Windows®95登場に備え、CPUには高速Pentium™プロセッサ(100MHz、90MHz、75MHz)。PCIバスアーキテクチャを全機種に採用。さらにプラグ&プレイ機能や、アプリケーションソフトのインストールを容易にする機能など先進性と拡張性を備えています。

■ビデオアクセラレーション機能、4倍速CD-ROMドライブを標準搭載。

なめらかな動画再生を実現するビデオアクセラレーション機能搭載のウィンドウアクセラレータ採用(PCI接続)。最大1,280×1,024ドット(256色)が表示可能、4倍速CD-ROMドライブの標準搭載、16ビットステレオPCM音源など、快適なマルチメディア環境を実現しています。

■ノートパソコンとの連携機能もさらに強化。

携帯型機とのデータ送受信のために赤外線通信インタフェースユニット(IrDA規格準拠)を増設可能な115,200bps対応高速シリアルインタフェースを標準装備。また、PCカードによるノートパソコンとの資産の共有を容易にする、前面アクセス可能なPCカードスロットアダプタや、フロッピィディスクドライブの追加内蔵が可能な、増設フロッピィディスクドライブベイも用意しています。

■優れたコストパフォーマンスを実現。

新98MATE Xシリーズは、先進の機能を身近にするとともに、98資産を活かしながら将来のシステムアップも安心して行える拡張性を両立。さらに優れたコストパフォーマンスも実現した、まさに次世代のマシンです。



intel inside pentium

ハイスペックを充実装備で
本体標準価格 **268,000円**(税別)から
※ディスプレイは別売です。

98MATE X 誕生

PC-9821 Xa10/C4 本体標準価格448,000円<税別> Pentium™プロセッサ〈100MHz〉
PC-9821 Xa9/C4 本体標準価格358,000円<税別> Pentium™プロセッサ〈90MHz〉
PC-9821 Xa7/C4 本体標準価格268,000円<税別> Pentium™プロセッサ〈75MHz〉
PC-9821 Xa10/C12 本体標準価格498,000円<税別> Pentium™プロセッサ〈100MHz〉
PC-9821 Xa9/C8 本体標準価格398,000円<税別> Pentium™プロセッサ〈90MHz〉
PC-9821 Xa7/C8 本体標準価格298,000円<税別> Pentium™プロセッサ〈75MHz〉
各モデルの内蔵HDD容量…C4:420MB、C8:850MB、C12:1.2GB

MS-DOS®6.2、Microsoft® Windows®3.1インストール済み。

※Microsoft、MS-DOSは米国Microsoft Corporationの登録商標です。Windowsは、米国Microsoft Corporationの米国およびその他の国における登録商標です。Windowsの正式名称はMicrosoft Windows Operating Systemです。PentiumはIntel Corporationの商標です。intel inside及びPentium Processorロゴはインテル社の商標です。Adobe、Adobe Photoshopはアドビシステムズ社の登録商標です。画面はすべてハメコミ合成です。

■各種ソフトウェアのアップデートサービス実施中
Windows3.1やWindowsNT、OS/2などのアップデートサービスを実施しています。詳しい情報はFAXサービスでお届けします。

■FAXサービス 東京:03-3552-9801 大阪:06-304-9821 (電話番号・FAX番号は、よくお確かめのうえおかけください。)
本広告の製品カタログと各種ソフトのアップデートサービス情報をFAXでお送りしています。①お手持ちのFAXから、ダイヤルしてください。②アナウンスに従ってBOX番号をダイヤルしてください。
7049 (PC-9821Xa情報BOX) 7403 (Windows3.1アップデートサービス第2回情報BOX) 7422 (Windows NT3.5アップデート情報BOX) 7421 (OS/2 J2.11アップデート情報BOX)

■新商品情報はPC-VANでも提供中!
PC-VANに接続したら、「J_NECPC」とキー入力してください。接続するには、PC-VANに加入している必要があります。加入については、PC-VAN事務局TEL 03(3454)6909にお問い合わせください。

<本広告に関するカタログを差し上げます。>ハガキに郵便番号、住所、氏名、年齢、勤務先を明記のうえ、〒108-01 東京都港区芝5-7-1 NEC 10-28560 98メイトX(トラ技コンピュータ6/17)係までお送りください。
<パソコンに関する技術的な質問・ご相談に電話でお答えします。>NECパソコンインフォメーションセンター 東京03(3452)8000 大阪06(943)9800 名古屋052(251)9800 東北022(262)9800 九州092(262)9800
受付時間…9:00〜17:00 月曜日〜金曜日(祝日を除く)電話番号は、よくお確かめのうえおかけください。

No.62

トラ技コンピュータ CONTENTS 1995年 7月号

PDPからIBM PC, PC98へと受け継がれる技術を理解しよう

特集 知っておきたいミニコン/パソコン変遷史

一般&連載

DOS/V & Windowsのページ

TRY COMPUTING

トレンド

特集について

アンケートはがきの内容から推測すると，本誌の読者の大半がミニコンを知らないか，知っていても見たことがない方々ではないでしょうか．本特集は，そんな方々を主なターゲットとして構成してみました．これからパソコンがどのように発展していくかはわかりませんが，より「高速化」されることだけはまちがいなさそうです．VAX11 の 100 倍のスピードをもつパソコンを持てる今，パソコンを何に使うべきか，再考してもよいのではないでしょうか．

次号のお知らせ　7月18日発売

特集　Windows 用デバイスドライバの仕組みと VxD

Windows のデバイスドライバは，本来 Windows が管理できるように仮想デバイスドライバ VxD として作成する必要があるのですが，ネットワーク管理などが充分に行われなかった Windows 3.1 ではその必要性がそれほど問題になりませんでした．しかし，これから登場する Windows95 では必須の技術になります．次号では，それをやさしく解説する予定です．

編集部から

＊8 ビットから 16 ビットへの移行があったように，現在は 16 ビットから 32 ビットへの移行期に入っている．前回は Z80 から 8086 という異なるハードウェアへの移行だったが，今回はハードウェアの移行ではなくソフトウェアの移行である．そのために，ユーザーにはゆるやかな変化に見えるが，アプリケーションを供給する側にとっては，全く新しい環境に対応するよりも難しくなっているように思える．〈山〉

＊国内でメモリ増設カードが入手できないという理由で，3 年前の C 社の DOS/V マシンが，二束三文で売っていた．雑誌の広告を見て，FAX で見積りをとると翌日に，$800，$700，$550 という三社から価格と送料が届いた．いちばん安い $550 の店に FAX で注文をすると 4 日目に自宅に届いて，消費税 1400 円を払った．日本円で 47,000 円(12 M バイト)，割高感のない値段だった．〈ヨ〉

＊ Delphi の英語版が発売されました．長い間 Pascal 系言語とは御無沙汰していたので，慣れるまで時間がかかりましたが，使ってみるとなかなかいいツールです．Visual Basic をはじめとして Visual ナントカツールがたくさんリリースされてきたので，サンデープログラマの方々も何を選択すればよいのか迷っているのではないかと思いますが，Delphi はおすすめです(本誌で連載を予定しています)．〈©〉

＊ Made in K××××のトレーナー，表示に色落ち注意とあるので，最初は 1 枚だけで洗濯．2 回目はもう大丈夫だろうと，他の物と一緒に洗濯機へ．なのに，他の物の仕上がりもきれいなブルー……．今どき，そんなに色落ちなんてしないのに，二度目までも色落ちしてくれるなんて！ 安物買いがかえって高くついた．少なくとも，表示は正しかったワケだけど，悔しい．〈T〉

トラ技コンピュータ 1995 年 7 月号

発行所 CQ 出版株式会社

〒170 東京都豊島区巣鴨 1-14-2
☎ 03-5395-2124(編集)/03-5395-2141(営業)/03-5395-2132(広告)
発行人 蒲生良治　編集人 山岸誠仁
1995 年 7 月 1 日発行
振替 00100-7-10665
定価は裏表紙に表示してあります
印刷/製本 三共グラフィック㈱ 表紙デザイン ㈱アイドマ・スタジオ トビラ・カット MINO MIED イラスト 神崎真理子

Printed in Japan

ICM

鮮やかであってこそ、薔薇。

16bit CPU搭載の鮮明フルデジタルコントロール画像。「薔薇」の文字さえにじまない。

A rose is a rose is a rose. バラはバラそのままの姿であってこそバラ、なのです。
自然の美を人工的に再現することは、近づくことはできても、決して到達できない永遠の目標なのかもしれません。
より自然な美を表現するために、バラがバラであるために、「鮮やかさ」の再現性はディスプレイ装置にとって欠かすことのできない資源です。
そのために搭載された16bit CPUは、フルデジタルコントロールで色だけでなく「薔薇」の文字さえにじまない鮮明画像表示を実現しました。
バラは主張します。その花弁は赤色ではなく、まさにバラ色であることを。

17インチマルチスキャンディスプレイ

RS series

NEW

「RS-A785N」「RS-A770N」は、PC-9800・Macintosh・DOS/V対応の17インチマルチスキャンディスプレイです。高輝度・高画質のアパーチャグリル管採用のCRTは、AGピッチ0.25㎜・最大解像度1600×1200ドット(RS-A770Nは1280×1024ドット)。表面処理にはARコートを採用し、鮮やかで高精細な画面表示を実現しています。また、16bitCPUを搭載し、画像の補正を自動でデジタルコントロールします。さらに、多彩なユーザーコントロール機能を備え、より細かく使用環境に合わせ込むことが可能なヘビーユースの要求にも応える高品質・高性能・低価格ディスプレイです。

【RS-A785N製品特長】

- AGピッチ0.25㎜・最大解像度1600×1200ドットの17インチアパーチャグリル管を採用
- オンスクリーン表示による多彩な調整機能
- 最大32種類の入力信号に対し、解像度プリセット記憶が可能（工場出荷時に、12種類プリセット済み）
- VGAからEWSまで広範囲のモードに対応
- VESA基準DDC規格(DDC1及びDDC2B)に対応
- プラグ&プレイを実現し、Windows95にも対応
- 電源ONの状態で作業を中断した場合、電力を約8W以下に低下させる、米国EPA Energy Star準拠のパワーセーブ機能搭載
- 信号コネクタは、BNCとDサブ15P(ミニチュア)の2系統を装備

■メーカー希望小売価格

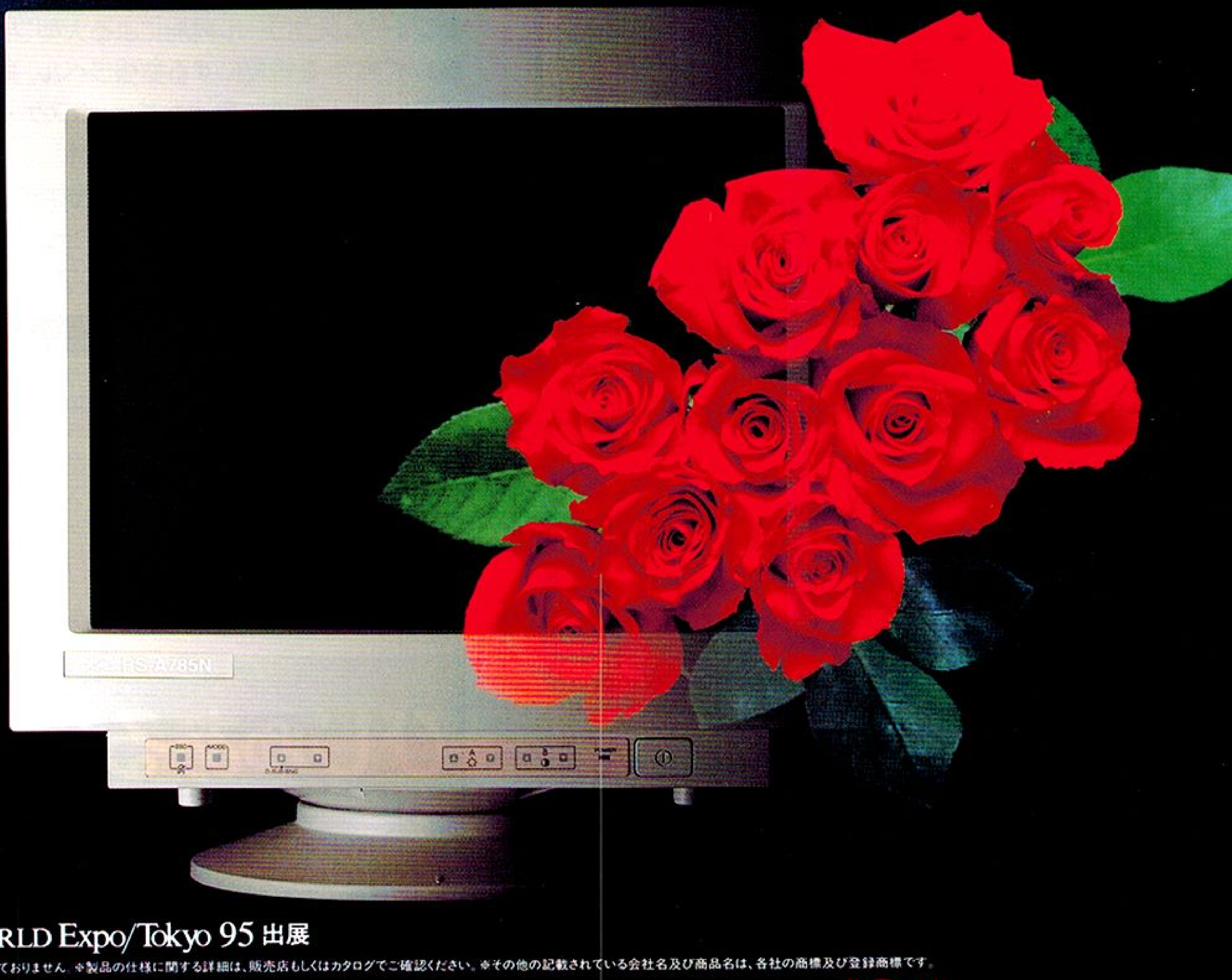
RS-A785N …¥128,000 付属品：DOS/V用接続ケーブル、98用変換コネクタ

【RS-A770N製品特長】

- AGピッチ0.25㎜・最大解像度1280×1024ドットの17インチアパーチャグリル管を採用
- 最大18種類の入力信号に対し、解像度プリセット記憶が可能（工場出荷時に、10種類プリセット済み）
- PC-98のノーマルモードからMacintosh21インチモードまで、幅広く対応
- VESA基準DDC規格(DDC1及びDDC2B)に対応
- プラグ&プレイを実現し、Windows95にも対応
- 電源ONの状態で作業を中断した場合、電力を約5W以下に低下させる、米国EPA Energy Star準拠のパワーセーブ機能搭載
- 信号コネクタは、Dサブ15P(ミニチュア)本体直結

■メーカー希望小売価格

RS-A770N …¥108,000 付属品：PC-98用変換コネクタ※DOS/Vパソコン用ケーブルは本体直結



WINDOWS WORLD Expo/Tokyo 95 出展

※表示価格には消費税は含まれておりません。※製品の仕様に関する詳細は、販売店もしくはカタログでご確認ください。※その他の記載されている会社名及び商品名は、各社の商標及び登録商標です。

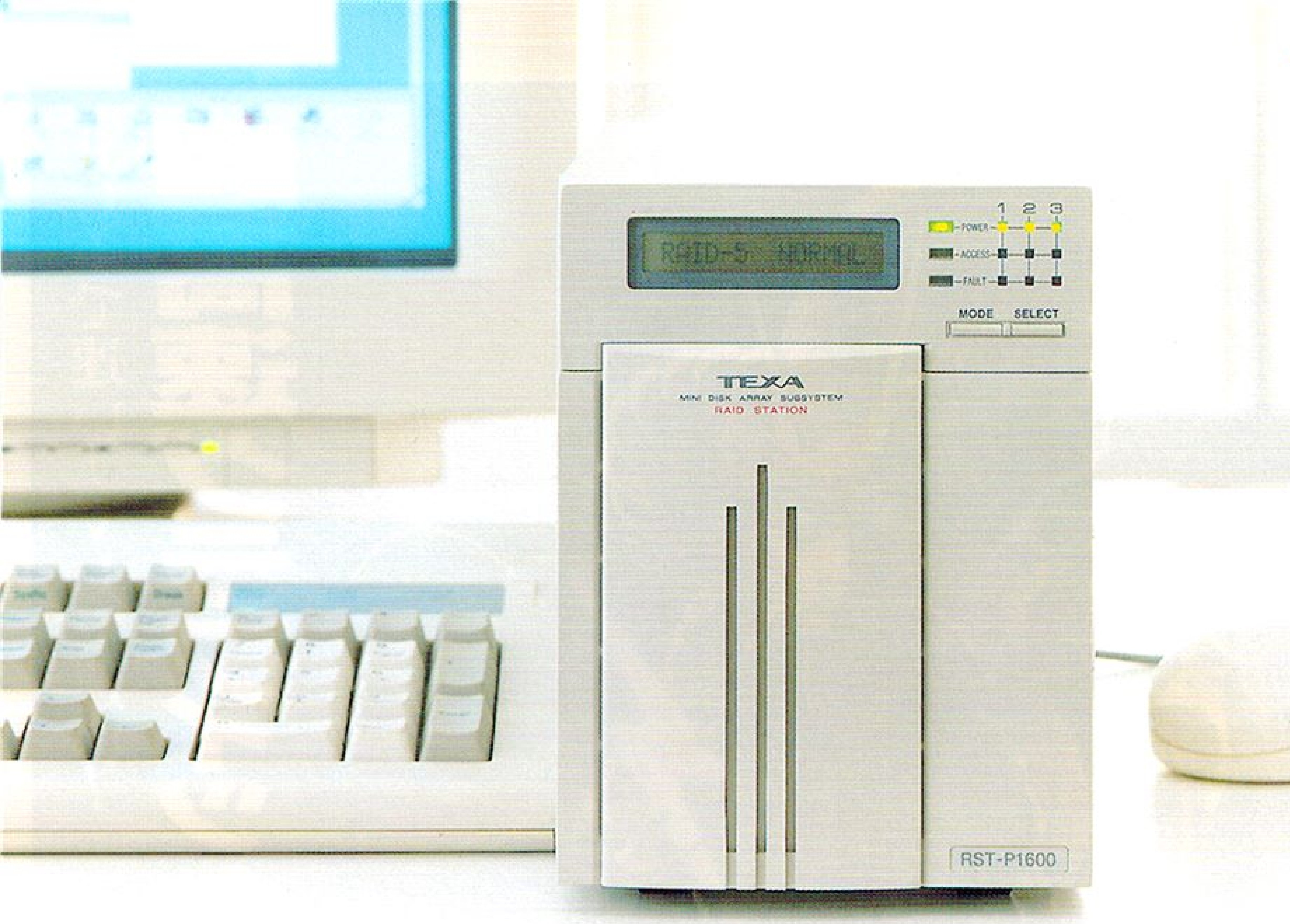
東京ショールーム ●〒101 東京都千代田区外神田1丁目15番16号秋葉原ラジオ会館6F ●TEL.03(5294)6570 ●FAX.03(5294)6575 ●定休日(月曜日・祝祭日) ●営業時間(AM.10:30～PM.6:30)

ICM 株式会社アイシーエム

●本社/〒532大阪市淀川区宮原3丁目5番36号新大阪第2森ビル7F TEL.06(350)5701 FAX.06(350)5711 ●東京支店/〒101東京都千代田区神田錦町2丁目9番地集美堂安田ビル6F TEL.03(3291)7493 FAX.03(3291)7494 ●名古屋営業所/〒450名古屋市中村区名駅南2丁目14番19号住友生命名古屋ビル5F TEL.052(561)4927 FAX.052(561)3644 ●北海道営業所/〒060札幌市中央区北5条西6丁目2番地の2札幌センタービル19F TEL.011(210)4778 FAX.011(210)4766 ●福岡営業所/〒812福岡市博多区店屋町1番31号東京生命福岡ビル5F TEL.092(262)1529 FAX.092(262)1530

資料請求No.5

資料請求No.6



漢字Talk 7.5
Mac OS

さらに身近に、さらにコンパクトに。これが、パーソナルRAID（レイド）。

ディスクアレイの高速性と高信頼性をパソコン環境で利用できることを可能にし、発売以来好評を得てきたRAID STATIONシリーズに、本格的なパーソナルモデルが仲間入りしました。
20万円を切る低価格で、RAIDアーキテクチャーの高い安全性を提供する、ミニ ディスクアレイ「RST-Pシリーズ」。ますます身近になって登場です。

高速 3台のハードディスクにデータを分散して、同時に読み書きを行うから高速。

安全 3台中1台のハードディスクが故障しても、2台で動作が可能。また、動作中故障したハードディスクの交換ができ、自動的にデータの再生が可能。

簡単 接続や操作は、従来のハードディスクとまったく同じイージーユース。

〈アプリケーションによって、3つのRAIDモードが利用できます〉

■RAID-0　：データを分割（ストライピング）して記憶することにより高いデータ転送レートが得られるもの。
［対応アプリケーション］
ビデオ動画編集、グラフィックデザイン等

■RAID-3/5：ハードディスクにパリティデータを書き込み、故障が発生しても処理の続行を可能にしたもの。
［対応アプリケーション］
データベース、
各種ネットワーク等



＊ホットリプレース機能があります。

FAST SCSI-II/FAST WIDE SCSI-II（SCSI-III）

RAID STATION Series
レイド ステーション

NEW FAST SCSI-2対応

RST-P840	（ディスク容量：約 850MB）	￥198,000
RST-P1600	（ディスク容量：約1700MB）	￥248,000

〈RAID STATION Pシリーズ共通仕様〉

- 外形寸法：130（W）×177（H）×274（D）
- 添付品　：ハーフピッチSCSIケーブル、
  フォーマットユーティリティ/デバイスドライバ
- 対応機種：NEC製　PC-9800、9821シリーズ
  EPSON製　PC-286/386/486シリーズ
  Apple製　Macintoshシリーズ
  AT互換機（DOS/V機）　アダプテック社製インターフェースボード AHA-1542CF、AHA-2940、AHA-2940W、Bus Logic社製 BT-956C等で接続検証済
  ＊対応機種については販売元までお問い合わせください
- 対応OS　：NEC製　MS-DOS Ver5.0A、6.20
  Apple製　漢字Talk7/7.5
  Microsoft製　WindowsNT/3.5、DOS/V
  Novell製　NetWare3.12J
  各社UNIX 等

＊弊社ディスクアレイのスピードを最大限に引き出すために PCI SCSIカードとの併用をお薦めいたします。

FAST SCSI-IIモデル（キャッシュ512KB）

RST-540	（ディスク容量：約540MB）	￥248,000
RST-1000	（ディスク容量：約　1GB）	￥298,000
〈PC-98用インターフェースボード付きモデル〉		
RST-540S	（ディスク容量：約540MB）	￥268,000
RST-1000S	（ディスク容量：約　1GB）	￥318,000

FAST WIDE SCSI-II (SCSI-III＊)モデル（キャッシュ1MB）

RST-540W	（ディスク容量：約540MB）	￥278,000
RST-1000W	（ディスク容量：約　1GB）	￥328,000
RST-2000W	（ディスク容量：約　2GB）	￥498,000
RST-4000W	（ディスク容量：約　4GB）	￥698,000

Lock付モデル登場（キャッシュ1MB）

RST-2000WL	（ディスク容量：約2GB）	￥508,000
RST-4000WL	（ディスク容量：約4GB）	￥708,000
RST-F4000WL	（ディスク容量：約4GB）	￥798,000

＊Pタイプ（68P）コネクタを使用しています。

7,200回転の高速ハードディスク搭載



キャッシュ容量 2MB

RST-F4000WL

■製造元■
日本テクサ株式会社
〒224 神奈川県横浜市都筑区佐江戸町384番地

■販売元■
株式会社テクサ インターナショナル コーポレーション

※価格には消費税は、含まれておりません。
※デザイン仕様等は、改良のため予告なしに変更することがあります。

〒224 神奈川県横浜市都筑区佐江戸町681番地　TEL.045（935）2560 FAX.045（933）2510
〒542 大阪府大阪市中央区南船場1-15-5万代ビル・順慶町2F　TEL.06（267）0666 FAX.06（265）5091
●技術的なお問い合わせ・ご相談は――テクニカルサポート窓口☎045-934-3583





国内初！64ビット・ドラゴンチップ採用。
衝撃の高性能＆ハイコストパフォーマンスを実現。

●ベストセラー「GAシリーズ」で培った日本語対応の実績と、安心の国内製造・販売・サポート。

●クラス最速！最新64ビットチップ〈トライデント社製DRAGONチップ〉搭載。低価格でありながら、超高速を実現したグラフィックアクセラレータボード。

●最大4Mバイトの画像メモリを搭載（2Mバイト版も4Mバイトまで増設可能）。1677万色表示で最大1024×768ドット、256色表示で最大1600×1200ドットに対応。

●DOS/V対応Windows3.1用ディスプレイドライバ、DOS/V対応Windows NT3.5、OS/2 WarpV3用ディスプレイドライバを添付。

●AVIファイル再生が速い！Windows3.1ディスプレイドライバは、DCIに対応。

●これまでの常識を打ち破る、クリアな画面表示を追求した高画質設計。

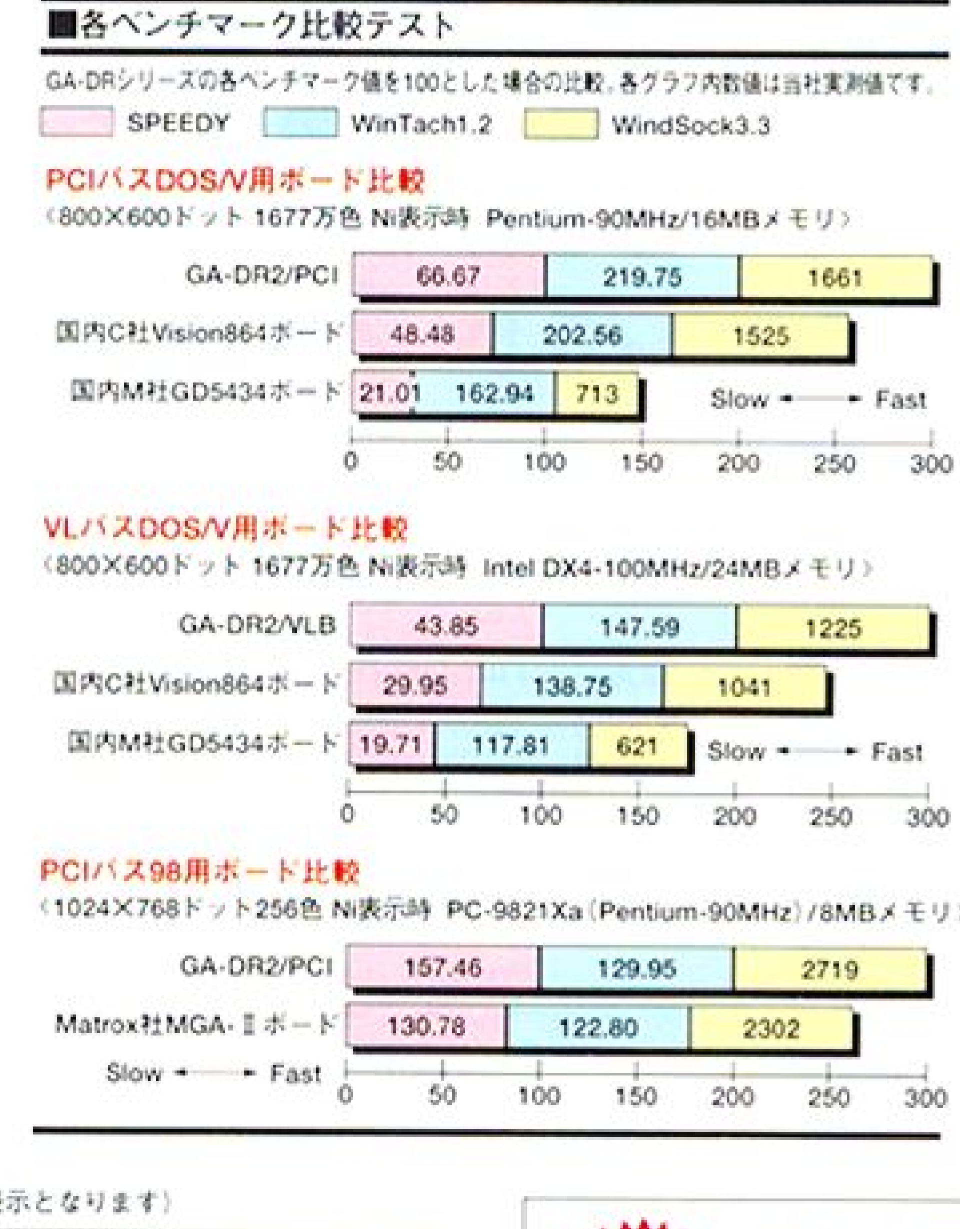
●VESAフィーチャーコネクタ搭載。

●ダブルプラットフォーム対応。（PCIバスを搭載した98&DOS/V）※

※PCIバス搭載のPC-9821シリーズ（PC-9821Xt,Xa,Xf）でご使用の場合は、別売の接続キットが必要です。

■各ベンチマーク比較テスト

GA-DRシリーズの各ベンチマーク値を100とした場合の比較。各グラフ内数値は当社実測値です。



グラフィックアクセラレータボード

DRAGON STAR64

for PCIバス

GA-DR2/PCI（2Mバイト版）……………¥29,800
GA-DR4/PCI（4Mバイト版）……………¥43,800

for VLバス

GA-DR2/VLB（2Mバイト版）……………¥27,800
GA-DR4/VLB（4Mバイト版）……………¥39,800

オプション品

■PCIバス搭載PC-9821シリーズ用接続キット
GA-KiT-98PCI ……………¥3,900
（接続ケーブル、サポートソフト、取扱説明書）

■2Mバイト版用2MB増設画像メモリ
GA-EXDR-2M ……………近日発売

■対応表示モード（2MB版もオプション増設メモリにより4MB版と同等表示となります）

解像度		640×480		800×600		1024×768			1280×1024			1600×1200
最大表示色	2MB	1677万色		1677万色		65536色			256色			―
	4MB	1677万色		1677万色		1677万色	65536色		65536色	256色		256色
走査方式		Ni	NiFF	Ni	NiFF	i	Ni	NiFF	i	Ni	NiFF	i

（i:インタレース／Ni:ノンインタレース／NiFF:ノンインタレースフリッカーフリー）

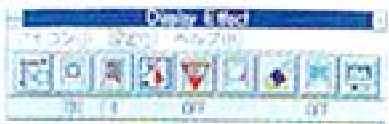
新発売

標準バス搭載 PC-9800シリーズ用
GA-DR2/98（2Mバイト版）………¥27,800
GA-DR4/98（4Mバイト版）………¥39,800

●Windows3.1,NT3.5,ジャストウィンドウVer.2,Lotus1-2-3R2.4J用各ドライバ添付
●機能拡張用セカンドバスTYPE II対応コネクタ搭載

高機能な標準添付アプリケーション

ディスプレイ特殊効果ユーティリティ
Display Effect



マルチメディア指向グラフィックソフト
SUPER KiD for Windows

Memory&Multimedia
アイ・オーは今 —— Memory & Multimedia
株式会社 アイ・オー・データ機器
本　社：〒920 石川県金沢市桜田町24街区1　TEL.0762-60-3355　FAX.0762-60-3350
営業所：東京・大阪・名古屋・福岡
ショウルーム：I-Oマルチメディアプラザ（秋葉原ラジオ会館7F）　TEL.03-3258-1010　FAX.03-3258-1065

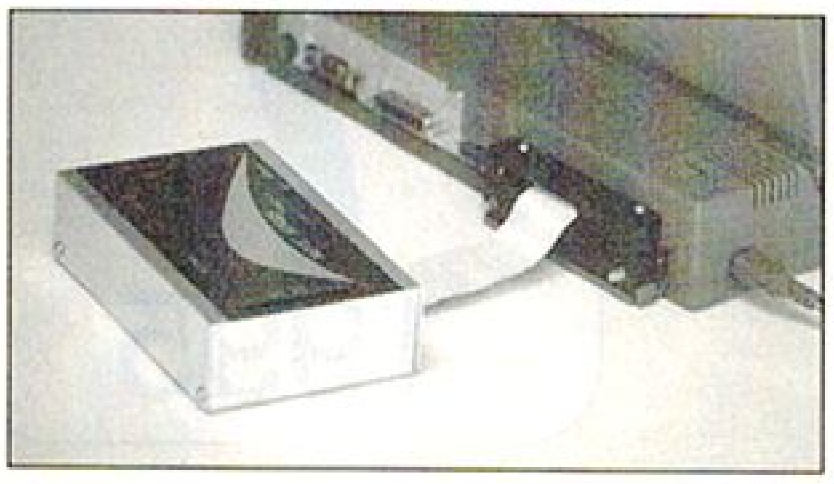
●新製品に関するお問い合わせは新製品案内窓口まで
本社／TEL.0762-60-1024　東京／TEL.03-5256-1024　大阪／TEL.06-368-1010
［TEL受付時間：祝・祭日を除く 月〜金曜日／午前9:30〜12:00 午後1:00〜5:00］
●自動FAXサービス（24時間）ガイダンス方式　東京／FAX.03-3254-9600
※ガイダンス方式ですので、プッシュトーンの出るFAXをご使用ください。

ECN
ELECTRONICS & COMPUTER NEWS

7
1995

この欄に掲載されている情報は各企業からの広告メッセージです.
新製品、話題の製品、キャンペーン製品などをダイジェスト・スタイルでご案内しています.
ぜひご活用ください.
なお、掲載情報へのお問い合わせは、各々の綴じ込みリーダーズ・カードをご利用になるか、
直接広告主にご連絡ください.

ノートPC用計測＆制御モジュール
NOTE-WINGS



◆ノートPC98（拡張バス110P）用の小型マルチI/O。Lotus123等と組合せて使用のデータ収集処理用サンプルソフト他付
◆A/D.12bit×8ch, D/A.12bit×2ch
Dio.32bit,CT.2ch,EXP-I/OバスCN付
◆ケーブル接続型 NTML-98B ¥48,000
(拡張バス直接固定型 NTML-98D¥43,000)

㈲ローム理研
〒431-04 静岡県湖西市新所原2-6-7
TEL 053-572-3961　FAX 053-577-1969

資料請求No.9007

Ni-Cd・Ni-MH電池６本用
急速充電器FRC－601

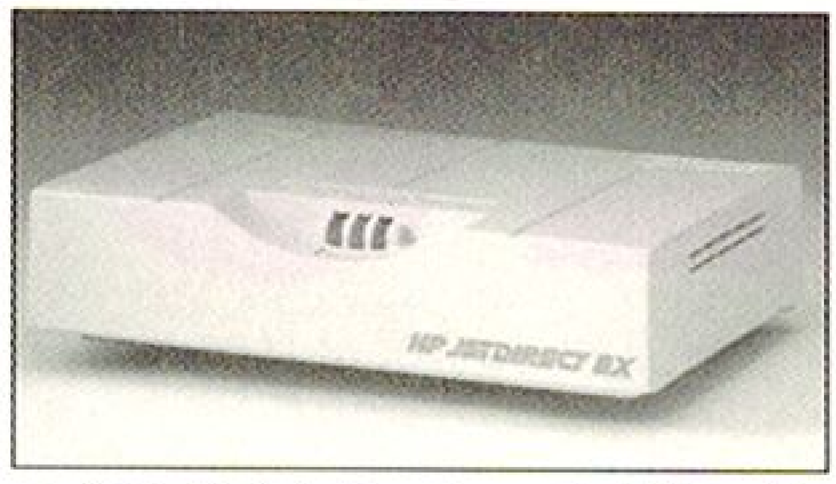


単Ⅲ型ニッカド・ニッケル水素電池6本を約１時間で急速充電《定価:12,800円》
◆－ΔV制御・スイッチモード充電を採用
◆電池温度・タイマーによる保護機能
◆メモリー効果を解消する放電機能付
◆小型軽量・高効率・操作簡単・低価格
ACアダプタ、シガライタケーブル、コンディショナーなど別売カスタム製品もOK

インタープラン㈱　TEL 03-5261-9307
〒162 新宿区新小川町5-20

資料請求No.9008

ネットワーク・プリントサーバ
ＨＰ社　JETDIRECT EX



■パラレルインターフェースを持つプリンタをネットワークへ直結
■ＮｅｔＷａｒｅ，ＷｉｎｄｏｗｓＮＴに対応
■高速化・共有性・設置の自由度を実現
●J2382B（イーサネット用）　69,000円
●J2383B（トークンリング用）79,000円

日商エレクトロニクス(株)PCプロダクト部
TEL 03-3544-8387　FAX 03-3544-8201
〒104東京都中央区築地７－３－１

資料請求No.9009

モニタ・キーボード共有システム
XrossKey98/AT,AT/AT



・２台のパソコンを１台のキーボード・モニタ・マウスで使用できます。
・パソコンの組み合せは98/ATとAT/AT。
・使い慣れたキーボード(98付属 IBM101, 106)が、そのまま利用できます。
・キーボード操作で切替が可能で、その他便利な機能色々。詳しくは下記まで。

㈱オブジェクト 豊中市刀根山元町12-27
E-mail info@object.co.jp
TEL 06-844-1747　FAX 06-844-1760

資料請求No.9010

DSP専門パワーシステムハウス
ディジタル信号処理

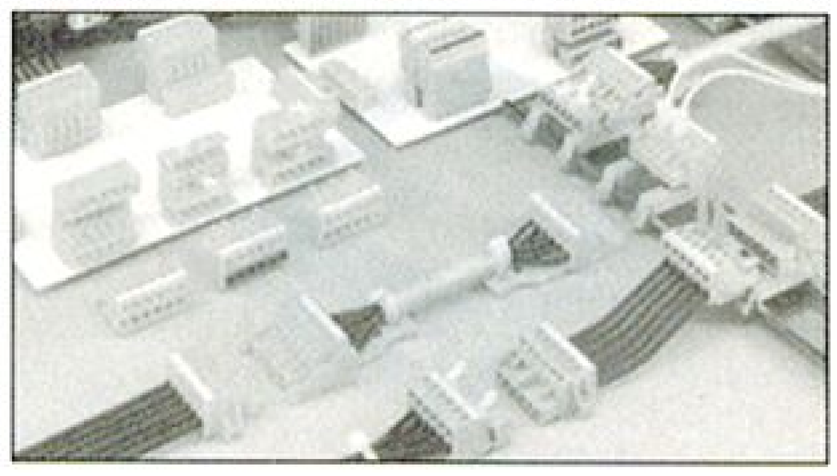
DSP
ソフト＆ハード

ＤＳＰソフト開発からハード開発・製造まで一貫して受注できます。ＤＳＰソフト開発単独のご依頼も歓迎いたします。
●音声処理＝分析・合成・認識・識別・ボコーダ・ＶＯＸ●音響処理＝雑音除去・ＡＮＣ・信号検出●伝送処理＝エコー消去・モデム・セキュリティ・誤り制御●画像処理●自動制御●計測●解析 etc

㈱スペクトラ　TEL 0272-63-1178
〒371 群馬県前橋市広瀬町1-29-17

資料請求No.9011

ケージクランプねじなし端子台
MULTI CONNECTION SYSTEM

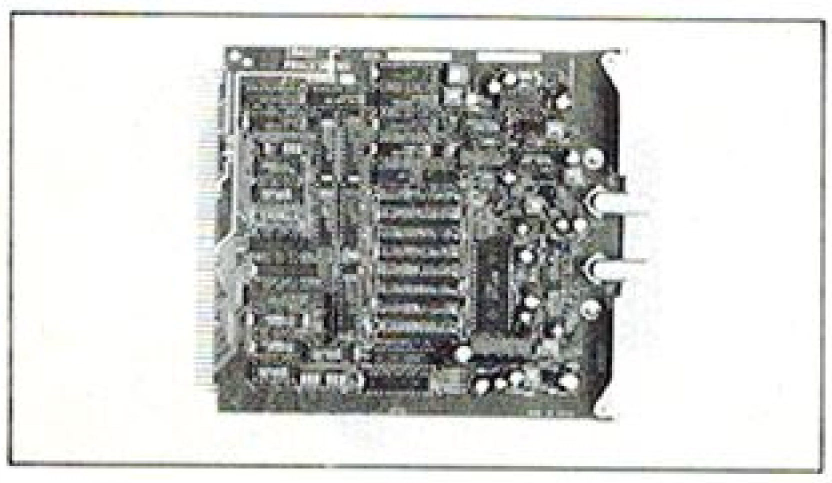


721／722／723／731シリーズ。
●誤挿入防止機構、嵌合ロック機構付きコネクタータイプが加わり更に電線の抜け防止用固定プレートも加わった。
●定格電圧：250V～400V　●定格電流：12A～16A●極間ピッチ：5，5.08，7.5，7.62mm

ワゴジャパン(株)
TEL 0423-30-7511　FAX 0423-30-7575
〒183東京都府中市日鋼町1-1 Ｊタワー

資料請求No.9012

フレーム・メモリ（モノクローム）
CT-9800B

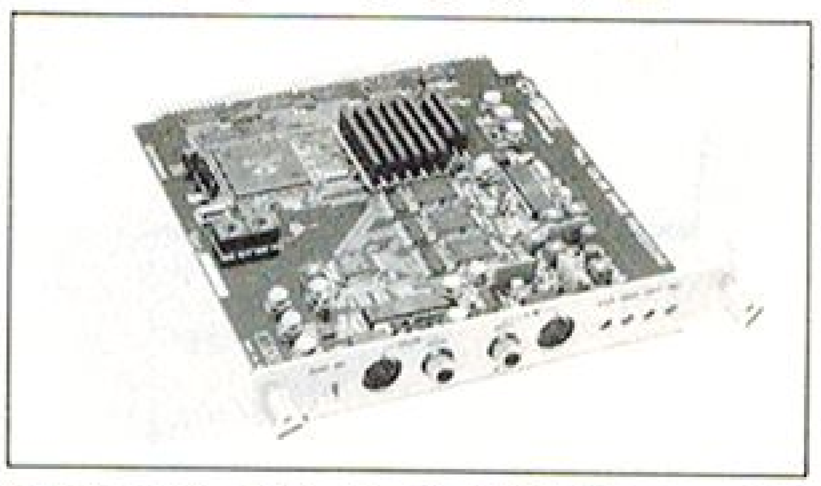


特長●1画面の記憶と表示出力：512(H)×512(V)×8bit 1画面 or 512(H)×256(V)×8bit 2画面●入出力は各1ケのBNCコネクタ●入力画面への計測用スケールや文字・図形のスーパーインポーズ●ダイレクトメモリマッピングによる高速で簡単なアクセス●対応機種：PC98、PC486、FC98●サンプルソフト付●オプション有

サイバーテック(株)　TEL 06-372-5558
〒531 大阪市北区本庄東3-9-15

資料請求No.9013

フルカラー・フレーム・メモリ
ＣＴ－９８０１Ｅ

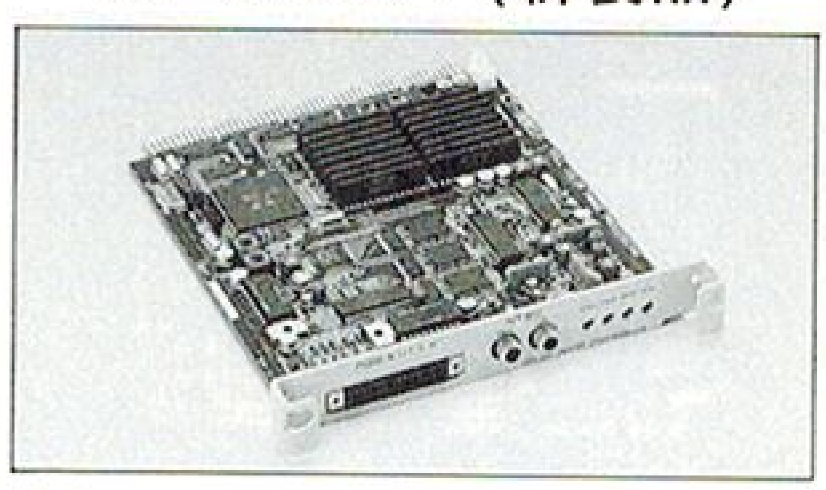


●シンプルデザイン＆高品質●フルカラー1677万色●ＸＹ座標で使い易い512(H)×512(V)画素を採用●入出力はNTSCコンポジット、Ｓ端子信号●アスペクト比1：1、4：3切換機能●1/2縮小機能●ダイレクトメモリーマッピング●CT-9801Cプログラムコンパチブル●PC-9801全機種対応●低価格

サイバーテック(株)　TEL 06-372-5558
〒531 大阪市北区本庄東3-9-15

資料請求No.9014

フルカラー・フレーム・メモリ
CT-9801F（新製品）

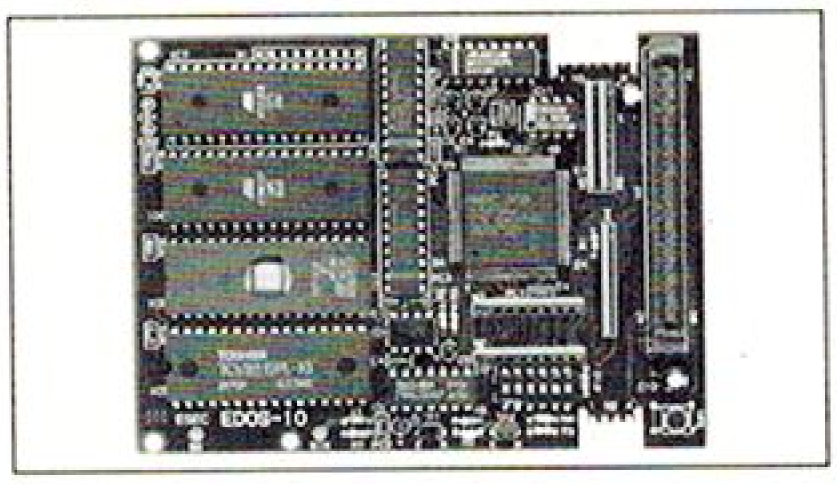


特長●1677万色フルカラー●XY座標で使い易い512(H)×512(V)画素で２画面、又は640(H)×512(V)画素で1画面●入出力：NTSC、S端子、RGB信号、PC画面へのスーパーインポーズ●ダイレクトマッピング（2Mバイトのプロテクトメモリー又は128KB×16のバンクメモリー、32ビットパックドピクセル）●PC98全機種対応

サイバーテック(株)　TEL 06-372-5558
〒531 大阪市北区本庄東3-9-15

資料請求No.9015

FA-LANサポート
組込みカードパソコン

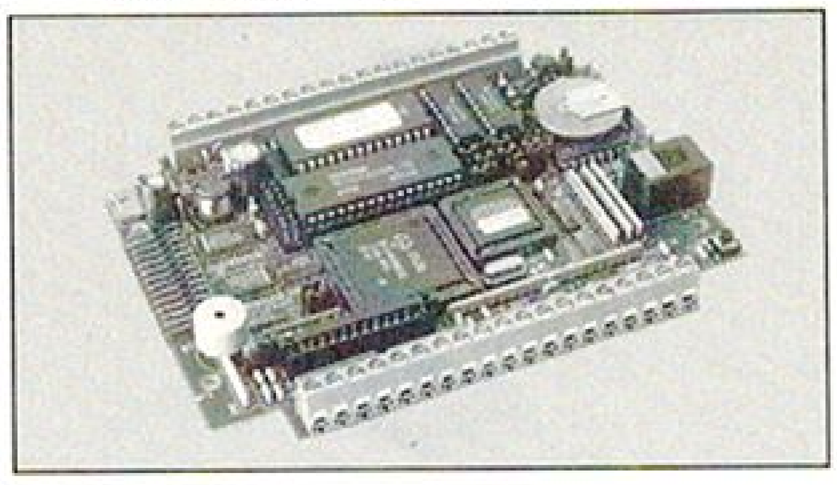


●アイス、ライタ不要　86×118mm
●FAに最適なトークンパッシングLANをサポート。パソコン1台で、各ノードのソフト開発を可能とするLANスコープを用意。
●ROM−DOS(データ・ライト社製)搭載、開発キット用ソフトE-TERM有。

日本エセック(株)
TEL 03-5256-3166　FAX 03-5256-3167
〒101 千代田区外神田2-10-8　富士ビル

資料請求No.9016

メカ制御からデータ収集まで！
C言語コントローラ



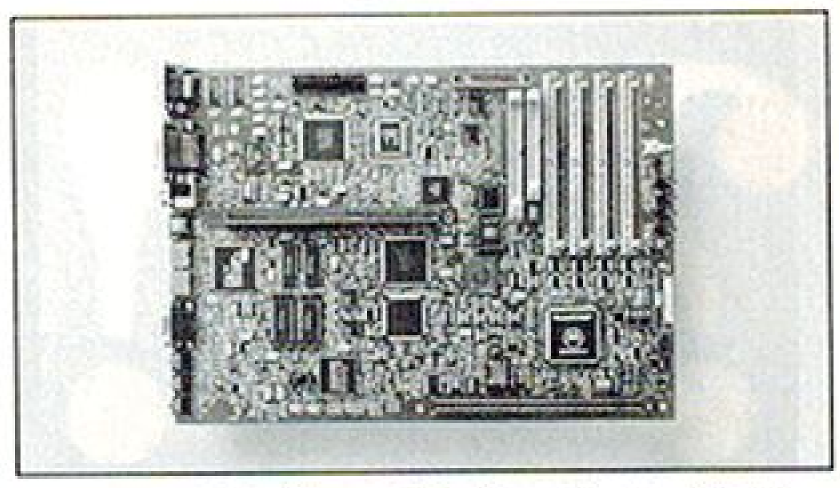
C言語コントローラは、シーケンサに代わりメカ制御ができ、アナログ制御や計測器からのデータ収集など組込み用途に応じて豊富な種類があります。

開発はWindows上で編集からデバッグまで快適にでき、ICEを必要としないのでとても低価格です。

カナース・データー㈱ SH事業部　〒101
千代田区神田佐久間町2-11-1笛木ビル3F
TEL 03-3851-2680　FAX 03-3851-4568

資料請求No.9017

PowerPC603/604搭載マザーボード
〈Ultra〉



高性能で低価格。量産組み込みに最適。
●PowerPC603(66MHz)/604(100MHz)●マルチOSサポート●ISA/PCI拡張スロット●SVGA(1280×1024)サポート●2〜128MB DRAM●シリアルポート×2●SCSI-2インタフェース●Ethernet●サウンドI/O●パラレルポート●マウス/キーボードインタフェース●5年間保証付き

丸文(株) 〒103 中央区日本橋大伝馬町8-1
TEL 03-3639-9896　FAX 03-3639-3727

資料請求No.9018

簡単・低価格・C言語用ライブラリ
Power Window



●簡単で低価格なC言語(MS-C,BCC)用の画面制御ライブラリです。
●同一のEXEがDOS/Vと98で動作します。
●約200個の使いやすく有用な関数を用意しています。
●導入日よりプログラムの開発工数を一気に短縮できます。
●価格　¥29,800（税別）マニュアル付

(株)パワーソフト世田谷区大原2-25-4-103
〒156　TEL 3327-3109　FAX 3327-9837

資料請求No.9019

NTTダイヤルインサービス端末用
電話交換シミュレータ

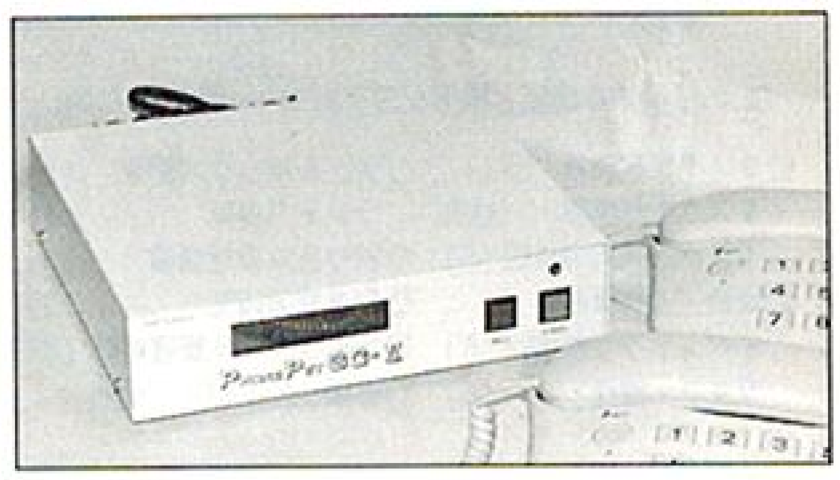


■ダイヤルインサービス端末を対象に相互接続および通信テストができます。
●DP(20PPS)･PBを自動識別 ●スイッチ切替でアナログ通信端末の試験も可能
●同機能内蔵FAX装置などの販促用途や開発ツールに最適 ◎NTT回線接続不可
●標準価格 ¥148,000(納期約2週間)

㈲アップロード
〒734 広島市南区西旭町14-25
TEL 082-254-8400　FAX 082-254-8777

資料請求No.9020

開発ツールや各種イベント用に
電話交換シミュレータ(1:1)

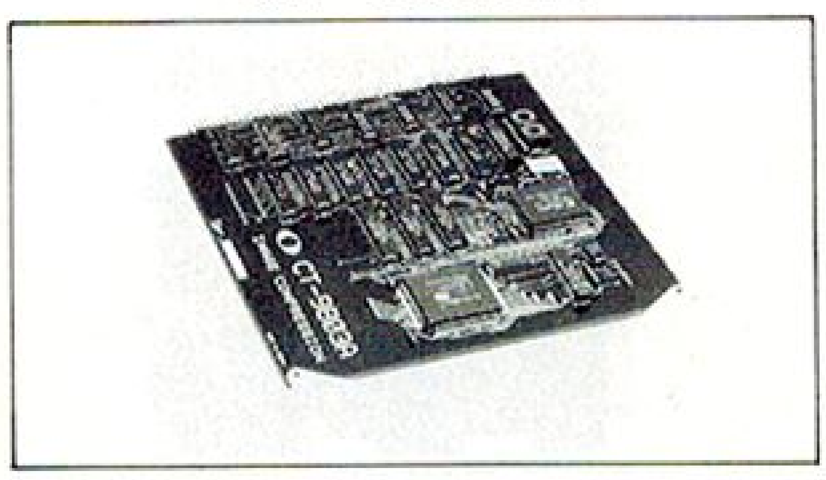


■電話機,FAX,モデム等アナログ通信端末相互の接続／通信テストができます。
●DP･PBを自動識別　●ダイヤル桁数を最大20桁まで設定可　●接続過程･ダイヤル情報を液晶表示　◎NTT回線接続不可　☆主な用途 TEL/FAX/PC通信のデモ 通信機器開発等　通常価格 ¥70,000 を税別特価¥38,000　(有)アップロード

〒734 広島市南区西旭町14-25
TEL 082-254-8400　FAX 082-254-8777

資料請求No.9021

画像圧縮・伸張ボード
CT-9803A

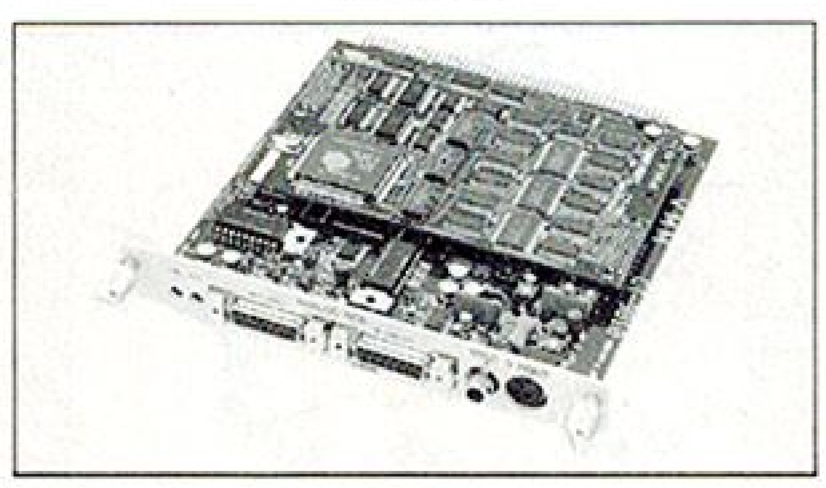


特長●JPEG(国際標準方式)採用
●1/25程度の圧縮で画像劣化が殆どない
●当社姉妹品CT-9801CおよびCT-9800Bをはじめ各社のフレームメモリ用インターフェイス・ソフト付
用途●画像ファイリング●画像通信●その他ディジタル画像の圧縮・伸張に最適

サイバーテック(株)　TEL 06-372-5558
〒531 大阪市北区本庄東3-9-15

資料請求No.9022

マルティメディア用ビデオボード
VMM

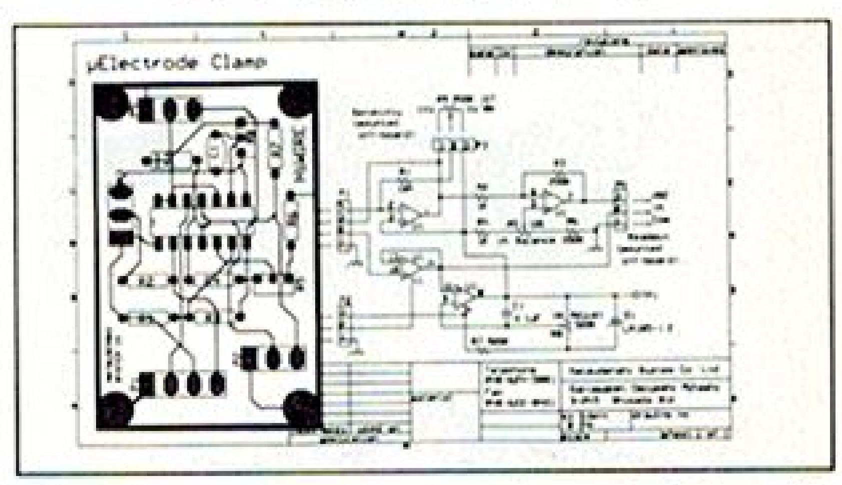


NTSCビデオ信号をPC-98シリーズのパソコン画面に動画表示。
●1677万色フルカラー動画表示
●オーバーレイ＆キャプチャ
●フルスクリーン・フルモーション録画再生(JPEG)
(用途) プレゼンテーション・画像通信・リアルタイム画像解析など

サイバーテック(株)　TEL 06-372-5558
〒531 大阪市北区本庄東3-9-15

資料請求No.9023

電子設計CADforNEC・IBM
HiWIRE Ⅱ



From USA WINTEK, High grade& Easy !!
回路図・プリント基板、両方の設計機能

Electronic Design Tools	¥140,000
Autorouter for DOS	¥180,000
Autorouter for WINDOWS	¥240,000
Data Viewer with Site	¥70,000

Leaflet, NECorIBM Demo Disk送呈，御要求は，FAX.048-622-8401まで．

松電子システム(株)　TEL 048-624-5881
〒331埼玉県大宮市三橋3-245島田ビル1F

資料請求No.9024

企画・開発・設計・試作
総合エンジニアリング

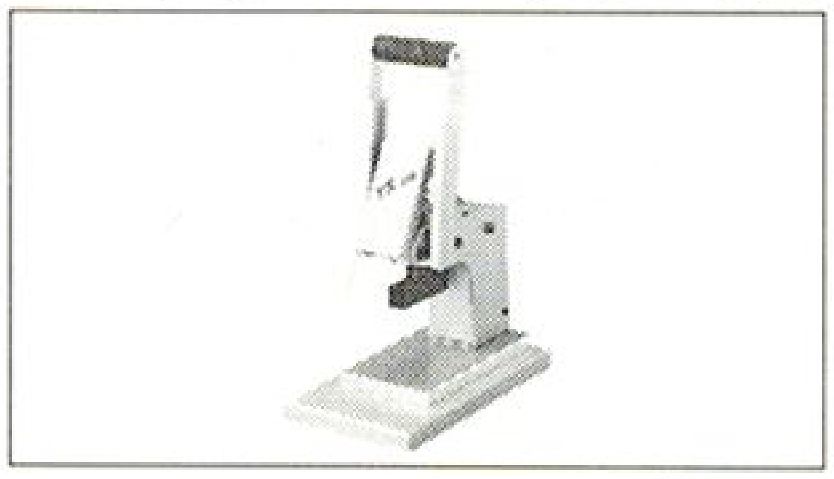
? アナログ回路設計
デジタル回路設計
メカトロニクス
ソフトウエアプログラミング !

設計／開発は外注できます
アイディア／企画の具体化
■AV製品の総合設計
■無線通信機器の総合設計
■計測ロボットの開発
■メカトロニクス機器
■特注計測器／制御機器

(株)ケント電子設計
TEL 03-3495-0861　FAX 03-3495-0493
〒153　目黒区目黒2-11-11 HKビル3F

資料請求No.9025

圧接工具のスグレモノ
ハンドプレス　ＹＳ－114



フラットケーブルコネクタ圧接用ハンドプレス　ＹＳ-114は、各種メーカーのコネクタを圧着できます。コネクタ固定板の調整で、様々な大きさのコネクタが圧着可能です。厳選された材料で、熟練した職人の手で製造されますので高品質・耐久性抜群の優れものです。

(有)吉村製作所
TEL 0489-24-1197　FAX 0489-24-4929
〒340 埼玉県草加市谷塚上町412-20

資料請求No.9026

試作・開発・量産向 高性能・低価格
巻線機　ＷＭ－０１



●マイコン応用の多機能型単軸巻線機で複雑な巻き方も楽にこなし、しかも作業者に極めて優しい静音設計です。
・主軸　　　５０００rpm
・トラバース巾　ＭＡＸ.９０mm
・電源　　　ＡＣ１００Ｖ(８０ＶＡ)
・重量　　　１２Kg

製造元　大井川電気製作所
TEL・FAX　０５４３－６５－５１１９
〒424　静岡県清水市横砂東町30-3

資料請求No.9027

H8/500シリーズマルチタスクＯＳ
ＲＯＭＯＳ

OSの提供する機能と標準システムコール数(カッコ内)

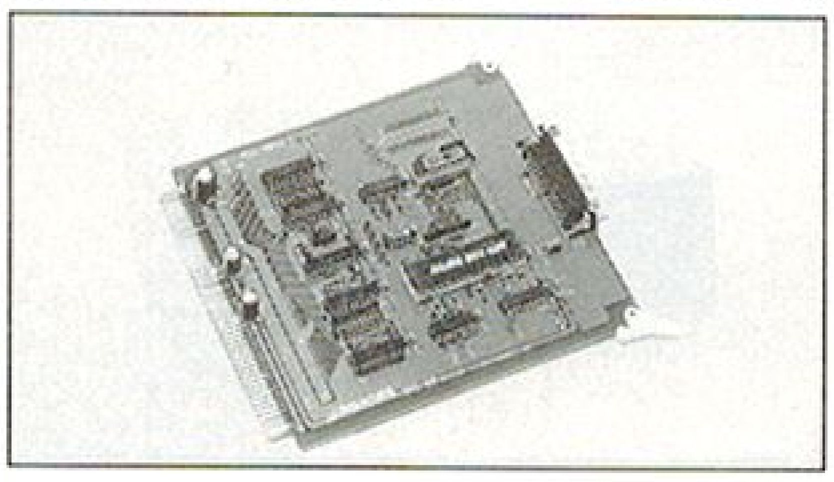
◎タスク管理機能		メールボックス機能	(5)
タスク管理機能	(13)	ランデブ機能	(4)
タスク付属同期機能	(9)	◎割り込み管理機能	(3)
◎同期／通信機能		◎例外管理機能	(3)
イベントフラグ機能	(6)	◎メモリプール管理機能	(4)
セマフォ機能	(6)	◎拡張SVCサポート機能	

●μＩＴＲＯＮ仕様に準拠
●容易なシステム構築
ＯＳの機能モジュールはライブラリ形式で提供
●高速なリアルタイムＯＳ
１０MHz動作時、割込みマスク時間１５μs、タスク起床時間４４μs
■　価　格　　26,000円～

(有)システムワン　TEL 0776-38-7290
〒910-37 福井県福井市浅水二日町403

資料請求No.9028

超高速，AutoFlow制御，98対応
RS-232C通信ボードSRS-98

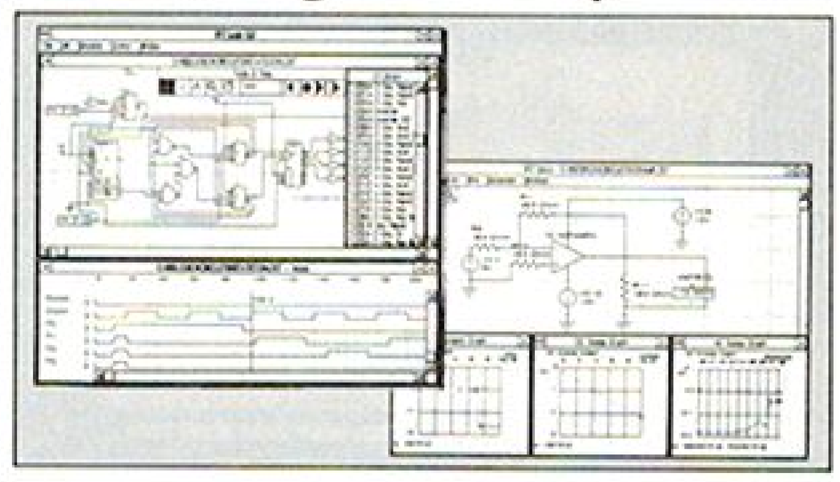


・75～576Kbps(1.2Mbps) ・RS-232C 1ch
・FIFO送/受各16byte ・AutoFlow制御
・PC98Cバス対応・¥9,800ー・AIR/CRAFT, CCT-98Ⅲ,WTERM,KTXなどの通信ソフトで使用可。・8M系の98でも19200bps以上の高速通信が容易。・14400や28800の高速モデムの必需品。・NiftyID:HHE02566メール可。

(有)ヒューマンデータ
〒567 大阪府茨木市西駅前町13-21-602
TEL 0726-20-2002　FAX 0726-20-2003

資料請求No.9029

ﾃﾞｨｼﾞﾀﾙ/ｱﾅﾛｸﾞ回路シミュレータ
B²Logic／B²Spice

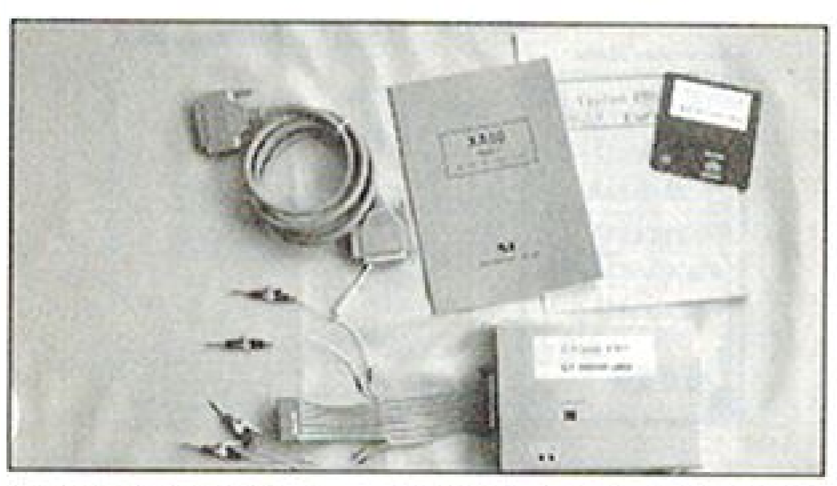


個人の研究者や技術者むけに開発されたシミュレータで研究者自身が個人で購入できるように適正な価格になっています。従来のこの種のソフトウェアに比べ大変安価ですが、設計できる回路の大きさやノード数など機能上の制限はありません。
B²Logic ¥39,800／B²Spice ¥29,800
●トラ技広告頁もご覧下さい●

(有)そらコンピュータ・プロダクツ
TEL 045-262-1097　FAX 045-232-1637

資料請求No.9030

Z80CPU用ROMエミュレータ型ICE
Z-VISION PRO

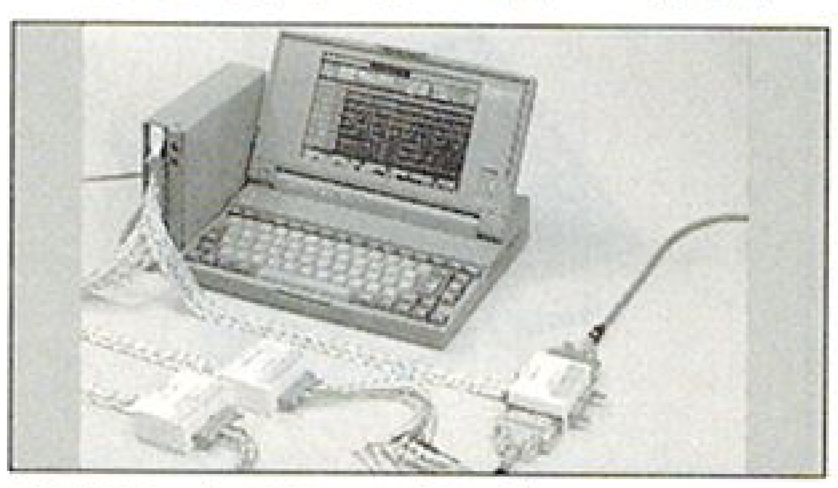


●ROMの内容（max.32kバイト）をそのままロードしてデバッグ可能●高速通信（最高76800bps）で快適操作●ROM空間，256バイトのI/O空間の全てをユーザに解放●NMIによる強制ブレーク機能付き
■対応ROM：27C64，27C128，27C256
■対応機種：PC-9801＆DOS/V両用
■価格：Z80版，KC5C8012版各¥69,800

(有)システムロード　TEL 048-475-0367
〒352 埼玉県新座市東北2-29-11

資料請求No.9031

ロジアナ＋ＲＳ２３２Ｃモニタ
ＬＯＧＩＣ－Ｚ　V3.4

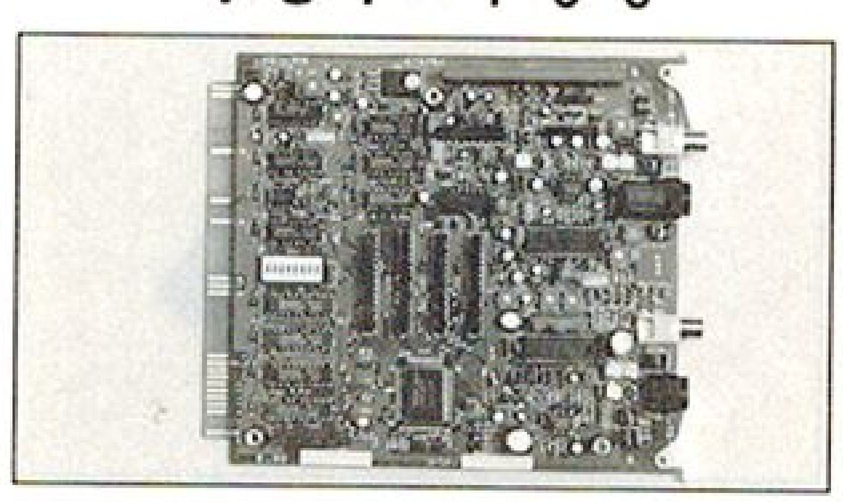


150MHz／16CH，30回／秒トレース
■ＰＣ98用・・・・・¥99,500
オプション
・450MHz／300MHzサンプリング（トリガー感度2.0ns）
・ＲＳ２３２Ｃモニタ／シミュレータ（内部76.8K、外部１倍 153KBPS）

(有)トップガン
TEL 0468-40-1020　FAX 0468-40-1021
〒239　神奈川県横須賀市岩戸5-8-20

資料請求No.9032

９８用スーパーインポーズボード
ＶＳＩ－１００

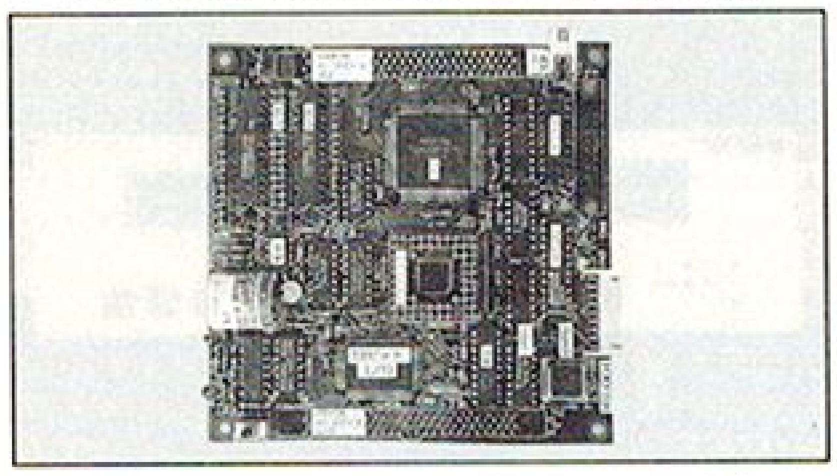


●NECのＰＣ－９８用拡張ボード
●７６８×４８０ドットの高解像度
●同時２５６色表示(Ｖ－ＲＡＭ 512Ｋ)
●コンポジット端子の他Ｓ端子も装備
●文字や図形の描画関数をライブラリとして付属（ＭＳ－C, Quick C 用）
●サンプルソフト(ソース)を付属

(株)システック
TEL 052-601-4911　FAX 052-601-5789
〒476 愛知県東海市名和町後西20-2

資料請求No.9033

68301＋72ビットI／O
機器組込みCPU基板

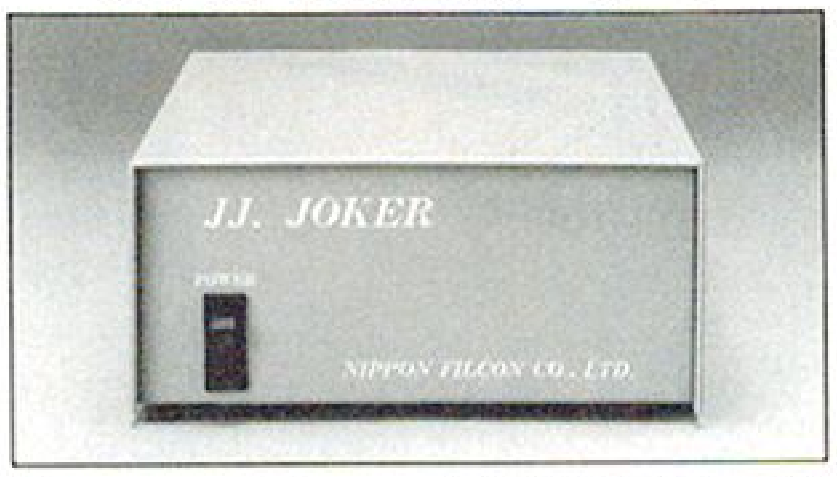


CPU TMP68301AF-16（15.9744MHz）
ROM 27C256×2 or 27C1000×2（未実装）
RAM TC55257×2（64KB）
RS232C MAX238（CPU内蔵1ch使用）
I/O外付72ビット＋CPU内蔵I/O
年月日時計、セントロ出力、電池
ICEソケット付（8/16MHz切替付）
120×120mm ¥58,000（消費税送料別）

(株)未宇研究所 TEL 0424-65-9444
〒188 東京都田無市西原町4-3-2-103

資料請求No.9034

汎用計測制御装置
JJ.JOKER E-1

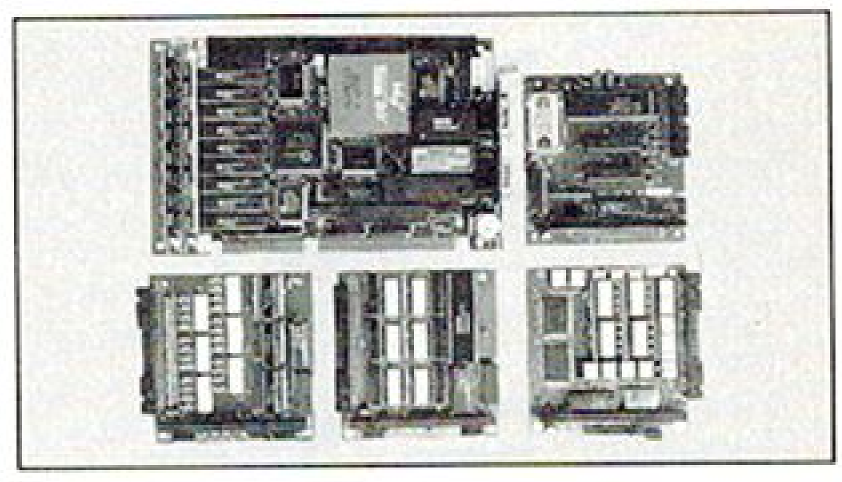


◆RS-232Cでパソコンと接続が可能。◆制御用コマンド内蔵により簡単操作。◆データロガー機能を内蔵し、データをパソコン上に保存可能。◆ドライバー1本で接続でき、容易にシステム構築が可能。◆アナログ入力:4点 出力:2点 デジタル入力:2点 リレー出力:2点◆計測機器、センサー類を豊富に用意。価格:¥97,000

日本フイルコン(株)〒206稲城市大丸2220
TEL 0423-77-1100 FAX 0423-77-5795

資料請求No.9035

DOS/Vマシンの組み込みに最適！
PC／104シリーズ

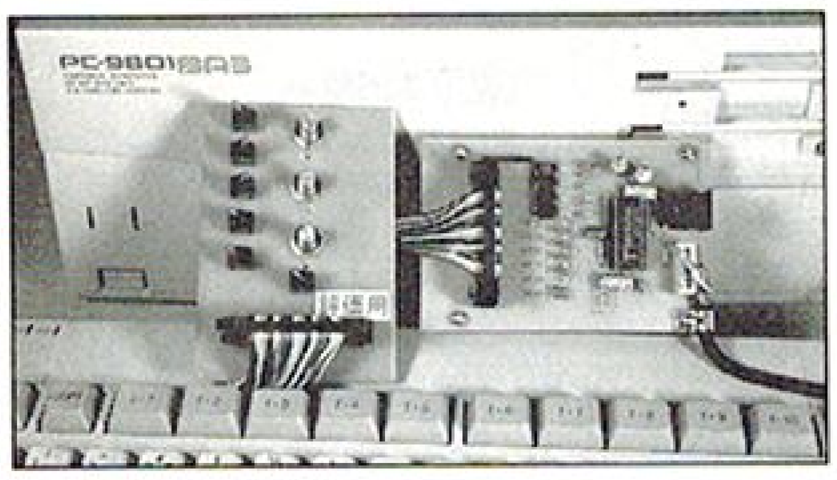


●PC/104シリーズはDOS/Vをカードゲージ、バックプレーン不用、限られたスペースに低コストでの組み込み可能●CPUカードは386SX、486LSC、80486DX2の選択可能●DOS/VだけでなくZ80、KC80SBCの拡張I/Oボードとしても使用し、柔軟にシステムを構築●ROMDISK、VGA、DIO、アイソレーション、A/D、D/A、モータCRL、ARCNET他

(株)ティーエーシー TEL 075-311-7307
〒600 京都市下京区西七条赤社町47

資料請求No.9036

プログラマへキーボード悩み解消
10KCV98VA

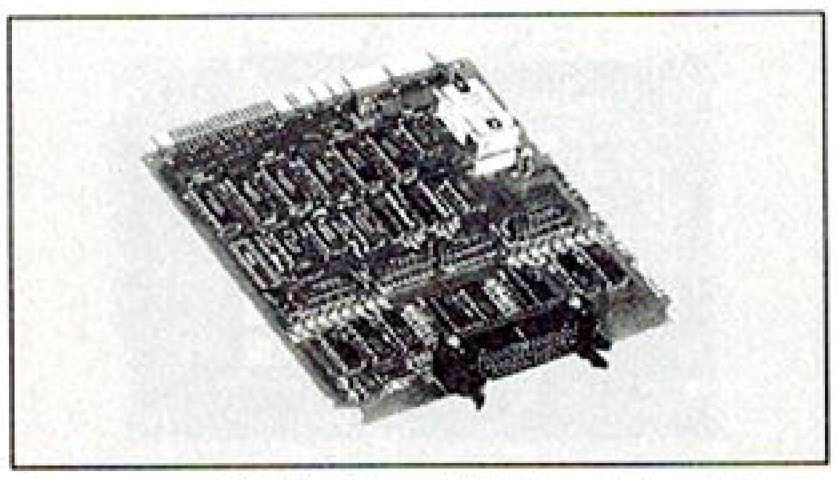


●98馴人のDOSVキーによるイライラ解消●PC98のLED付標準キーボードをDOSV OA DG 106キー（ミニDIN6P）へ中継する変換器●VM以前機械式ロック及親指シフト等特殊キーでは一部不具合有●通販1週間以内返却可、価格は税送料込 7,000円、住所氏名電話明記、振替+現金封筒同封による後払方式●販売店募集中、委託販売可

㈱テンショウ〒583羽曳野市白鳥3-4-3
TEL 0729-57-0789 FAX 0729-56-5159

資料請求No.9037

PC98用“破格”絶縁型I/Oボード
IO-16/16-ISO



●FA、LA、学習用途に最適な16bit 入力16bit 出力、絶縁型デジタル I/O ボード
●入力側に専用のDC/DCコンバータ搭載
●出力GNDを8bit毎に切離し可能
●ストローブ入出力各1点＆割込3点
●3mハーネス添付
■¥25,000 ■取扱店募集中

テクノ・グレース(有)
TEL 0770-52-7724 FAX 0770-52-7720
〒917福井県小浜市湊11-15-3

資料請求No.9038

パソコンセレクトシステム
doubles（ダブルス）



一人複数台のパソコン環境でキーボードマウス、モニタを共有します。
■7/1にリリース予定 定価各¥39,000-
・「doubles AT/MAC」(PC/AT MAC)
・「doubles PC/MAC」(98 MAC)
■その他製品ラインナップ
・doublesPC(PC98)、AT(PC/AT) ¥39,000-
・doubles Ⅳ/PC(PC98) ¥78,000-

㈱エアー
TEL 075-343-0750 FAX 075-343-0605

資料請求No.9039

不正コピー防止ツール
持出厳禁 V1.5



●コピープロテクトツール●Windows DLL呼出●DOSのCOM,EXEに直接プロテクト●強力キーフロッピ●手軽な運用キーハードディスク●ランタイムフリー●納品、販売、見本ソフトに最適●企業、学校、内部ソフトに使用可●LAN用開発中
■価格各48,000円 ☆ソフト開発承ります。データベース、DOS、Windows、Z80、86、ROM化

(株)ポトス TEL 078-753-6443
〒655 神戸市垂水区塩屋町4-1-7-303

資料請求No.9040

小型で低価格の普及型電力計
デジタル電力計

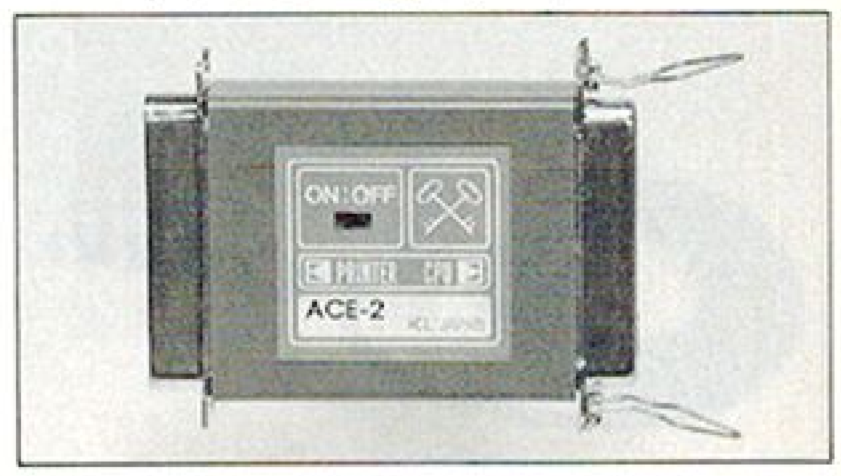


●コンセントに差し込むだけで面倒な配線が不要
●小型・軽量で取扱いが簡単
●ホールセンサー直接乗算方式で真の電力実行値表示
●その他デジタル電力計ファミリーもあります ●価格￥25，600～

(株)ユー・アール・ディー
TEL 045-502-3111 FAX 045-502-3632
〒230 横浜市鶴見区本町通4-169-3

資料請求No.9041

ソフトウェアのセキュリティ機器
「ACE」シリーズ

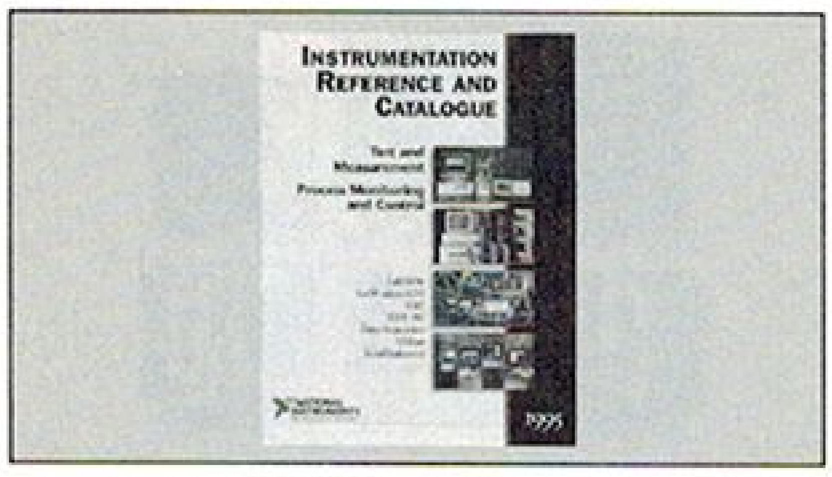


●セントロタイプ ACE－2 ●RSタイプACE－8 ●初期費用198,000円（書込機1台、ACE 20個）●買増分@3,800円～●ACE内に7bit9文字書込み可●チェック用ソースは契約メーカーに公開●独自の回路でNOTEでも安定動作●5個まで連結使用可能●BASICでも導入可●DOS, WINDOWS, WINDOWS-NT対応

(株)ICL® TEL 03-5285-0963 遠藤
〒169 東京都新宿区大久保2-4-15-4F

資料請求No.9042

コンピュータベース計測製品満載
1995年総合カタログ(無料)

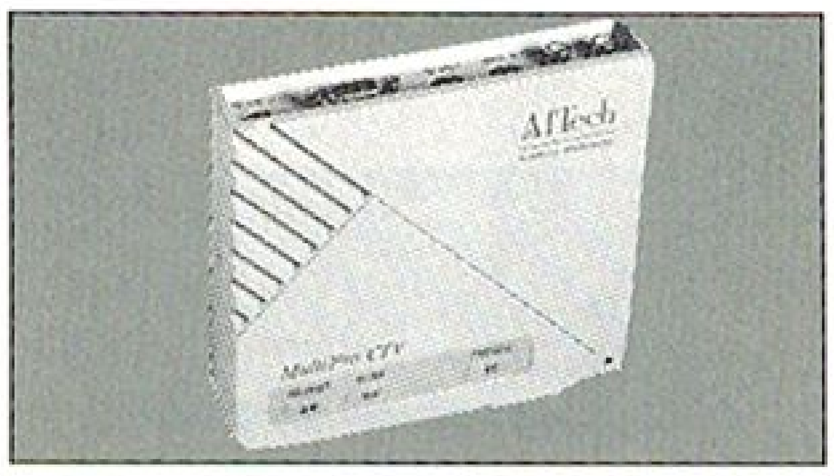


計測システム開発のためのソフトウェアとハードウェアを掲載しています。GPIB、プラグインデータ集録システム、信号調節アクセサリ、VXI、及びMXIについての学習に役立つ、包括的なチュートリアルが特徴です。資料請求カードをご利用ください。すぐにご送付します。

日本ナショナルインスツルメンツ(株)
〒142 東京都品川区戸越5-14-24 ITOビル
TEL 03-3788-1921　FAX 03-3788-1923

資料請求No.9043

パソコン画面がテレビでみられる
MultiPro　CTV

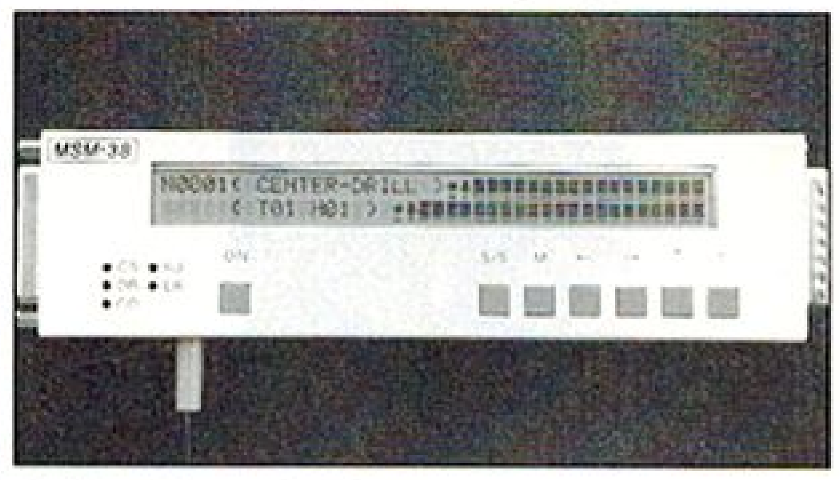


- ◆家庭用TV、大画面、ワイドテレビに適合
- ◆NEC/IBM/DOSV/Mac/Apple等すべてＯＫ
- ◆パソコン画面を録画、教育、展示、ゲーム
- ◆同機種中画面のクリアさは世界トップ
- ◆解像度640×480、16.7million色調対応
- ◆差し込むだけのアダプタ式、１年間保証
- ◆米国製、日本語マニュアル付、¥49,800

キャッスルＺ国際商事(有)　〒600京都市
TEL 075-361-2407　下京区中堂寺前田
FAX 075-361-8487　パレジット五条501

資料請求No.9044

超小型プロトコルモニター
ＭＳＭ－３８



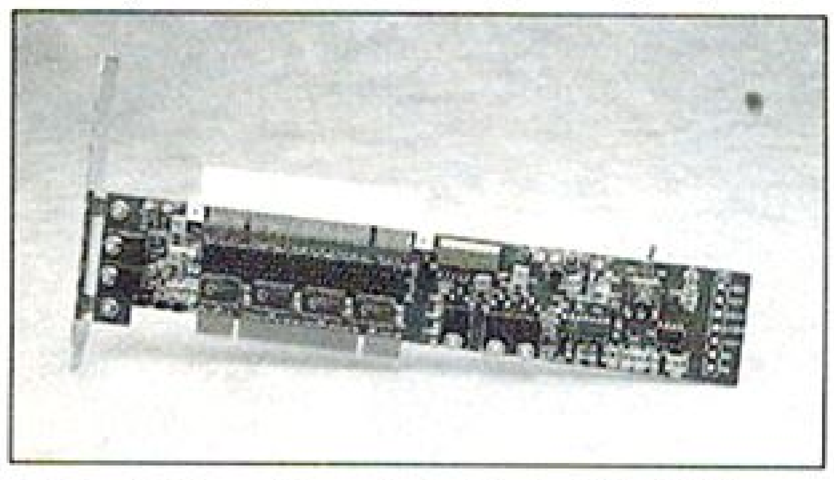
I/F：RS232C、RS422A、RS485
速度：非同期　300～19200BPS
表示：JIS、ASCII、HEX　ISO、EIA(NC)
　40×2行、記憶2560B(P･OFF保持)
電源：電池/AC100V オートパワーオフ
出力：保持データ出力

■本体価格　38,000円

マイクロサポート(株)　TEL 06-399-5577
〒532 大阪市淀川区宮原1-19-10

資料請求No.9045

ボード開発，検査，保守の必需品
PCI,ISA,VMEエクステンダ



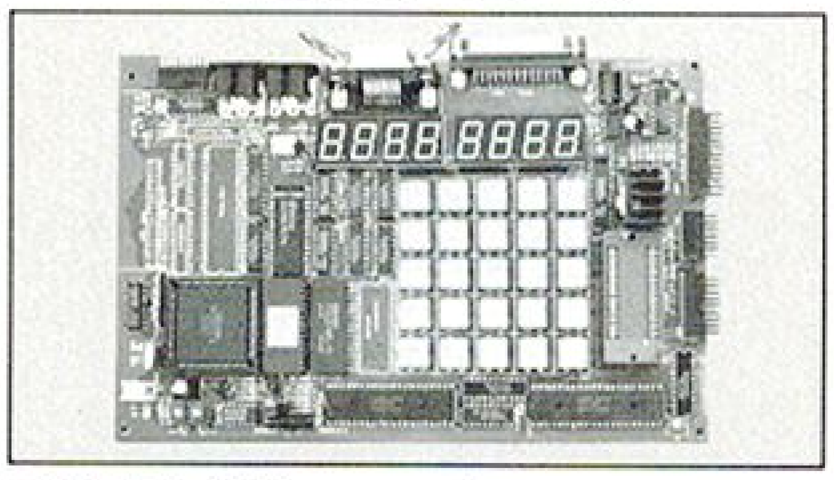
◎アイソレーションタイプ（端子付）
遅延250psの最新ICを用い，しかもPCの電源をOFFにせずにボード着脱可。
- ・PCI532，332(写真)・PCIX32－5,－3
- ・ISAIX16　・VLBIX　・VME100
- ・64bitPCI　・ＰＭＣ

◎ノーマルタイプ（プローブ端子付）
- ・PCIEX4－5H－D,－3H－D

立野電脳(株)　FAX 0428-77-7010
E-mail：GGC00374@niftyserve.or.jp

資料請求No.9046

作りながら学ぶマイコン入門基板
ＴＲＮジュニア



■豊富な機能：Z80系TMPZ84C013、ROM32K、RAM32K、25キー、8桁LED、セントロ、RS232C、A/D･D/A変換、スピーカ、ROMライタ、MIDI、120曲オルゴール、BASICおよびZビジョン対応。
■定価：基本キット12,000円、基本+拡張キット21,000円、完成品は+3000円

(株)ＥＳＰ企画 FAX 0774-65-1321(24時間)
TEL 0774-63-2743 (月～金AM10:00～12:00)
〒501-62岐阜県羽島市正木町三ッ柳123

資料請求No.9047

韓国エレクトロニクス技術誌
『電子技術』

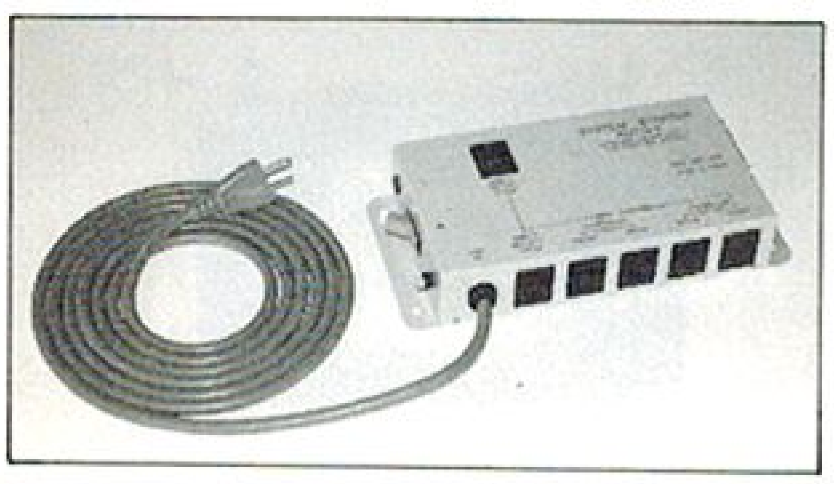


本誌は、ＣＱ出版社発行の「トランジスタ技術」と編集提携している韓国有数のエレクトロニクス技術誌です。いまや日本とともに世界の半導体市場で大きなシェアを占める韓国のエレクトロニクス分野で多くの読者を獲得しています。本誌のお問い合わせは下記まで。

(株)ダイナコインターナショナル
〒165 東京都中野区沼袋2-1-7-101
TEL 03-3389-1822　FAX 03-3389-1761

資料請求No.9048

パソコン通信革命
システムスターターAUTO-6

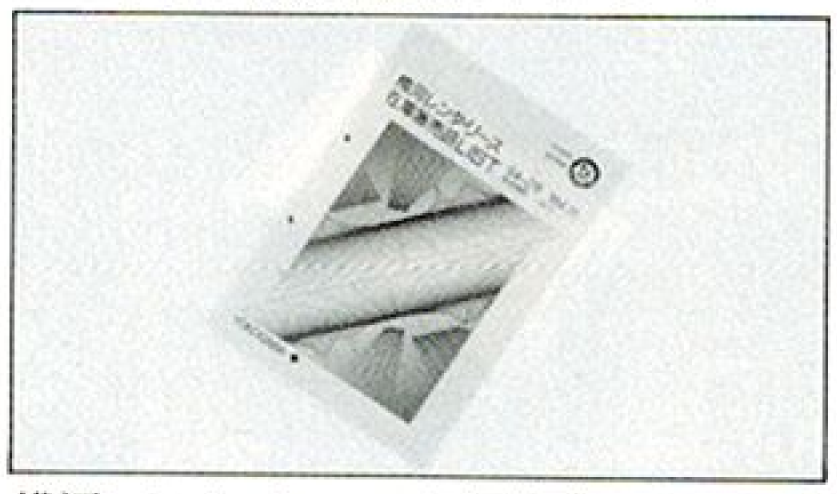


■電話の呼び出しベルが鳴ると自動的に出力コンセントがオンしてシステムを立上げ無人運転を開始、通信終了後は自動オフする。終了後も通信があったことをアクセスランプで表示。
■その他の機種として、CPUの電力消費を検出して周辺装置の電源を自動オン／オフするタイプとノートパソコン専用。

(株)エヌ・イー・エス　TEL 0286-82-8841
〒329-13栃木県塩谷郡氏家町氏家3488-4

資料請求No.9049

在庫物件をお手軽な価格でご提供
在庫販売品ＬＩＳＴ



横河レンタ・リースでは，保有している電子計測器やパソコン等のレンタル物件をリーズナブルな価格でご提供する「在庫品販売」を行なっています。ご予算不足の時、新製品を必要としない時などに有効です。高品質な製品をお届けする当社の「在庫品販売」をご利用ください。

横河レンタル･リース(株)
TEL 03-3343-0561　〒163-05 新宿区西新宿1-26-2　新宿野村ビル25F

資料請求No.9050

中国エレクトロニクス技術情報誌
ＥＤＮ Ｃｈｉｎａ



EBA・EDN Asiaに続いて、昨年の９月に創刊されたのがEDN Chinaです。中国本土だけをターゲットとしたこの媒体は、中国の主要都市に約20,000部配布されています。主な読者層は、エンジニアマネジメント及び、強い決定権を持つ工場長クラスを中心にしています。

(株)ダイナコインターナショナル
〒165 中野区沼袋2-1-7-101
TEL 03-3389-1822　FAX 03-3389-1761

資料請求No.9051

通常請求用

CQFAX資料請求システム

ECN専用ページ

「CQ (Convenient＆Quick) FAX資料請求システム」は、お忙しい読者にとり頼もしい情報収集方法です。手順は簡単。まず、このページをコピーして、空欄の5項目にご記入ください。あとはFAXで小誌まで送信するだけです。
いつでも何度でも、すぐ請求できる「CQFAX資料請求システム」をどうぞご活用ください。

資料請求カード　ECN　1995年7月号

当社記入欄
1 VOL 0 3 7 0
2 受　付

カード有効期限：1995年7月末日迄
送付先は勤務先に限ります。

3 登録内容変更箇所No.

4 請求状況 2　1＝新規　2＝登録済　3＝登録内容の変更

5 会員番号 CQ
＊会員の方のみご記入してください。

6 フリガナ
7 姓　名　姓　名

8 生年月日　19　年　月　日

9 〒　都道府県　区市郡　区町村
10 勤務先住所

11 会社名
12 部署名

13 勤務先電話番号　市外局番　局　番　電話番号　(　)

14 役　職　15 所属部門　16 担当分野

17 業　種　18 従業員数

19 広告資料請求欄 請求No. の欄に資料請求番号を記入し、A. B. C. のご希望の項目に○をつけてください

希望項目＼請求No.	90	90	90	90	90	90	90	90	90	90
購入希望	A	A	A	A	A	A	A	A	A	A
説明希望	B	B	B	B	B	B	B	B	B	B
資料希望	C	C	C	C	C	C	C	C	C	C

希望項目＼請求No.	90	90	90	90	90	90	90	90	90	90
購入希望	A	A	A	A	A	A	A	A	A	A
説明希望	B	B	B	B	B	B	B	B	B	B
資料希望	C	C	C	C	C	C	C	C	C	C

ECN専用

ご利用の手順

①資料請求会員番号の確認
↓
②このページをコピーする
↓
③空欄の5項目に記入する
1.会員番号
2.ご氏名
3.生年月日
4.資料請求番号
5.請求希望項目
↓
④03-5395-2104に送信する

※利用資格
資料請求会員の方だけがご利用になれます。会員未登録の方は、綴じ込みのカードでまず登録を完了してください。
会員番号をご確認したい方は下記までご連絡ください。

※記載事項
5項目に記入いただけないと広告主に連絡できません。

FAX番号：03-5395-2104

●お問い合せは下記まで●

CQ出版株式会社 広告部広告営業課 資料請求係　TEL.03-5395-2132

通常請求用

ECN

ELECTRONICS & COMPUTER NEWS

PR Information 1995

ＥＣＮは，製品情報をダイジェスト・スタイルでご案内する広告情報です．そのために，いくつかの特徴をご用意しています．これは，ＥＣＮが単に情報を提供するだけではなく，読者の方々にも広告主の方々にも便宜を提供することを基本趣旨としているからです．そのいくつかをご紹介します．読者側からみたメリットとしては，①定型原稿なので情報が見やすい②定型原稿なので検索しやすい③製品の特徴が簡潔にまとめてあるので把握しやすい④ＦＡＸ情報ＢＯＸでさらに詳しい情報を入手できる⑤資料請求システムを利用できる，などとご支持いただいています．一方，広告主側からみたメリットとしては，①低価格な広告料金②簡単な手続き③ぎりぎりまで繰り下げた進行日程による使いやすさ④統一形式による公平な広告情報⑤資料を即座に提供できるＦＡＸ情報ＢＯＸシステム⑥便利な資料請求サービス・システムなどをご支持いただいています．今後ともＥＣＮにぜひご注目ください．

低コストで３誌に掲載できる広告 ＥＣＮ

ECN

エレクトロニクス・エンジニア向けにはトランジスタ技術，ソフトウェアやシステム・エンジニア向けにはインターフェース，まじめなＰＣユーザ向けにはトラ技コンピュータと３誌に掲載できます．しかも，広告料金は次の通り見事なまでに低価格です．

4色1/9ページ・・132,000円
2色1/9ページ・・・99,000円
白黒1/9ページ・・・66,000円

資料請求No.9000　　BOX No#000

簡単な手続きで掲載できる広告 ＥＣＮ

ECN

一般的に，広告はさまざまな手続きを経て掲載に至ります．でもＥＣＮならば，わずかなステップの手順でＯＫです．申込から掲載までな次の通りです．

①広告申込書のご記入およびご提出
②所定原稿用紙へのご記入
③掲載写真のご送付
④広告料金のお支払い
⑤広告原稿の校正
⑥広告掲載

資料請求No.9000　　BOX No#000

締切日に余裕のある広告 ＥＣＮ

ECN

発行年月号	申込締切日	原稿締切日
95年３月号	１月17日	１月20日
４月号	２月17日	２月20日
５月号	３月17日	３月20日
６月号	４月17日	４月20日
７月号	５月17日	５月19日
８月号	６月16日	６月20日
９月号	７月17日	７月20日
10月号	８月17日	８月21日
11月号	９月18日	９月20日
12月号	10月17日	10月20日

資料請求No.9000　　BOX No#000

広告スペースは1/9ページに限定 ＥＣＮ

ECN

ペイド・パブリシティであるＥＣＮの基本原則の第一は，定型形式です．これは，見やすさ，検索しやすさ，内容把握のしやすさを考えれば，ご理解いただける範囲のルールです．その結果，大規模な企業から産声を上げたばかりの企業まで，同一のスペースで同一の規則に基づいた整然とした広告情報ができあがります．ＥＣＮは製品情報をお伝えするスペースに徹しています．

資料請求No.9000　　BOX No#000

ＦＡＸ情報ＢＯＸシステムもある ＥＣＮ

ECN

掲載してある広告情報を見て，すぐに詳しい資料を知りたくなるのが，読者の気持です．そんな気持にお応えするのがＦＡＸ情報ＢＯＸシステムです．
手順はとても簡単です．

①03-5570-1303にダイヤルします．
②メッセージに従って操作します．
③希望するＢＯＸナンバを入力します
④終了手続きをします．
⑤ＦＡＸで資料が送信されます．

資料請求No.9000　　BOX No#000

資料請求サービス・システム付き ＥＣＮ

ECN

掲載してある広告情報を見て，もっと詳しい説明を聞きたくなるのが，積極的な読者です．そんなときにお使いいただけるのが，資料請求サービス・システムです．手順はとにかく簡単です．

①資料請求会員番号を確認します．
②ＣＱＦＡＸ資料請求システムＥＣＮ専用ページをコピーします．
③空欄の５項目に記入します．
④03-5395-2104に送信します．

資料請求No.9000　　BOX No#000

TRY COMPUTING トラ技コンピュータ SPECIAL

1995年7月号

PDPからIBM PC，PC98へと受け継がれる技術を理解しよう

特集　知っておきたいミニコン／パソコン変遷史

パーソナルコンピュータは，集積回路となったマイクロコンピュータから発展してきたものですが，その発展の過程ではミニコンピュータの影響を強く受けています．

では，このミニコンとはどんなコンピュータだったのでしょうか．Unixや C言語が，このミニコン上で開発されたということはよく知られていますが，そのミニコンがどのような内部構成になっていたかなどはあまり一般に知られていません．

そもそもミニコンは，LSIや半導体メモリなどが登場する以前のコンピュータですから，そのハードウェア構成は現在のパソコンとはまったく異なるものです．現在と比べると，未熟な半導体技術の上に作られてはいますが，コンピュータに込められたアイデアや工夫には驚かされることが数多くあります．

また，その後のパソコンの発展の歴史の中でも，とりわけIBM PCの登場は世界標準パソコンが生まれる決め手となり，国内ではPC98シリーズが事実上の標準機となったので，独特のパソコン文化を持つことになりました．

本特集では，これらのコンピュータの変遷をわかりやすく解説しました．

C O N T E N T S

特集 目次

第1章

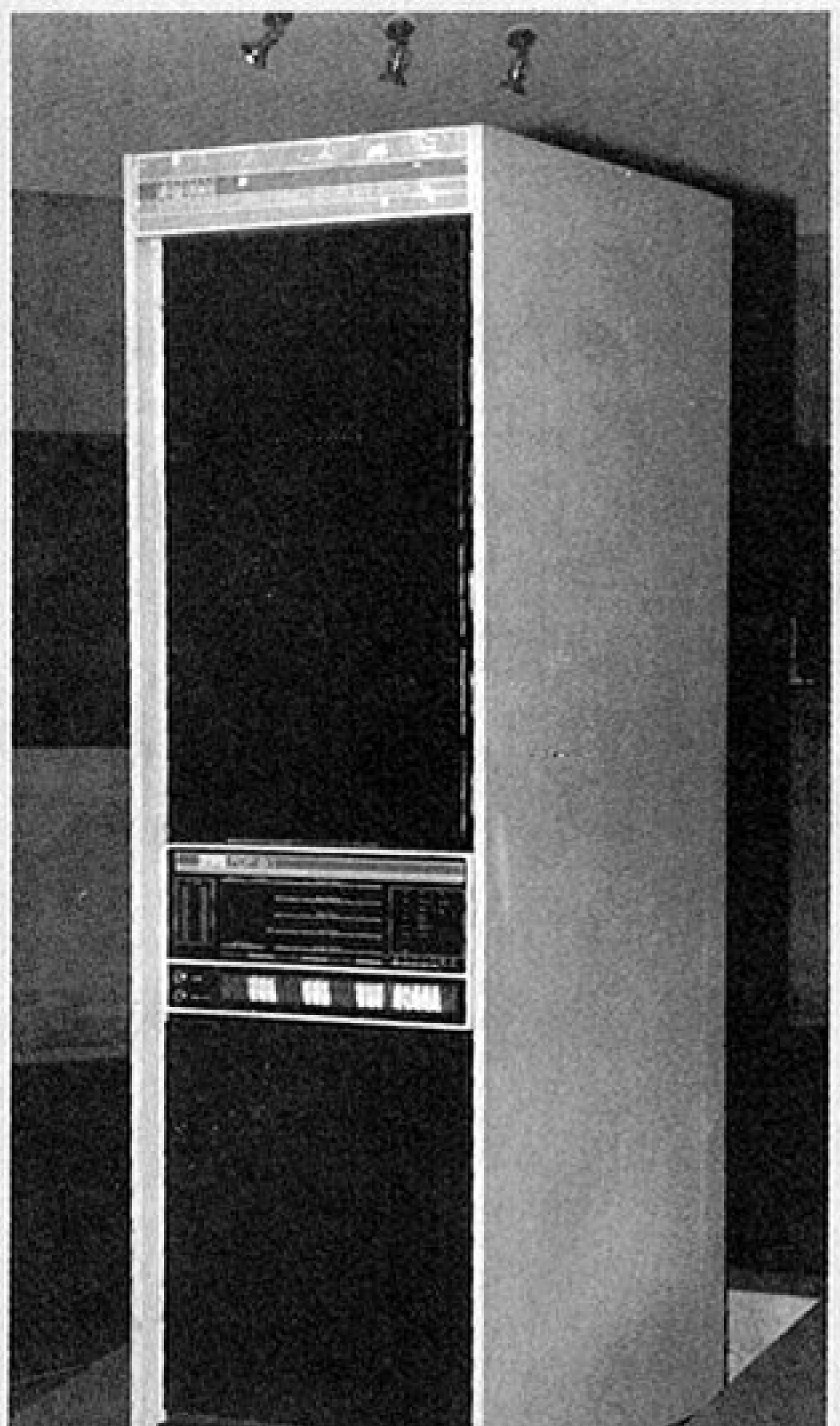
PDP-8, PDP-11, VAX11 とはどんなコンピュータだったのか

パソコンの礎になった DEC 社のコンピュータの変遷

勝男　厚

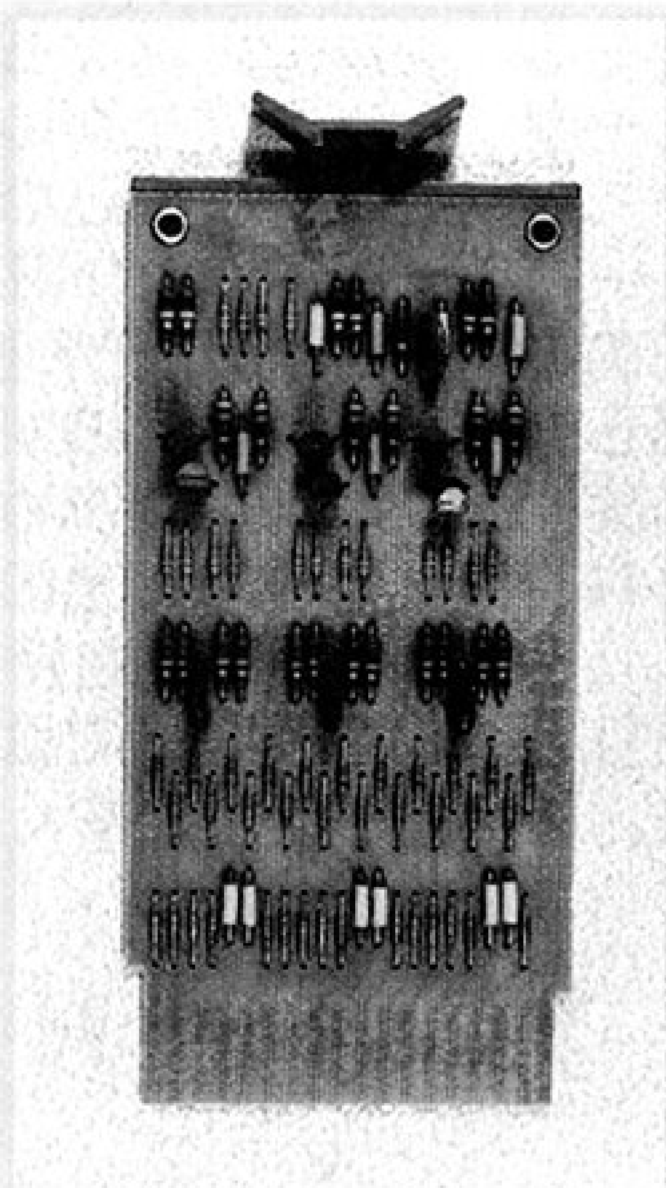
「ミニコンってどんなコンピュータだったの」．その素朴な疑問に答えようというのが，本章の主題です．そこで，手始めにミニコンの誌上博物館を作ってみました．

今のパソコンのほうが，遥かに高性能であるのは明白ですが，このようなコンピュータの上でC言語やUNIXなどが開発されたことを考えると，感概深いものを感じます．

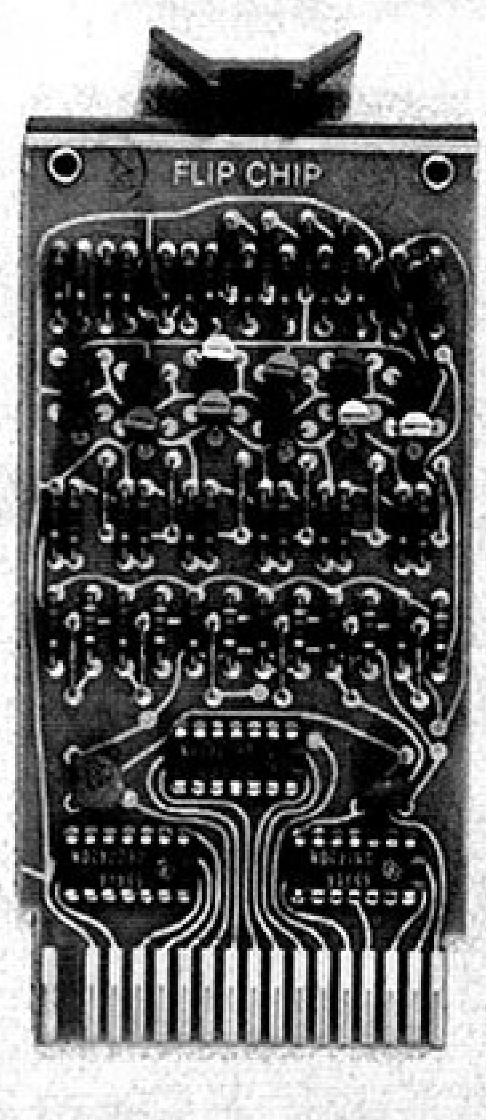


ワード長	12 ビット
最小メモリ	4 K ワード
最大メモリ	32 K ワード
キャッシュ	なし
サイクルタイム	1.5 μ 秒
アーキテクチャ	ワードマシン AC×1 I/O 命令あり
大きさ	高 0.8 m 幅 0.5 m 奥 0.6 m
重量(最小構成)	60 kg

〈写真 1〉 PDP-8 の外観と基本仕様

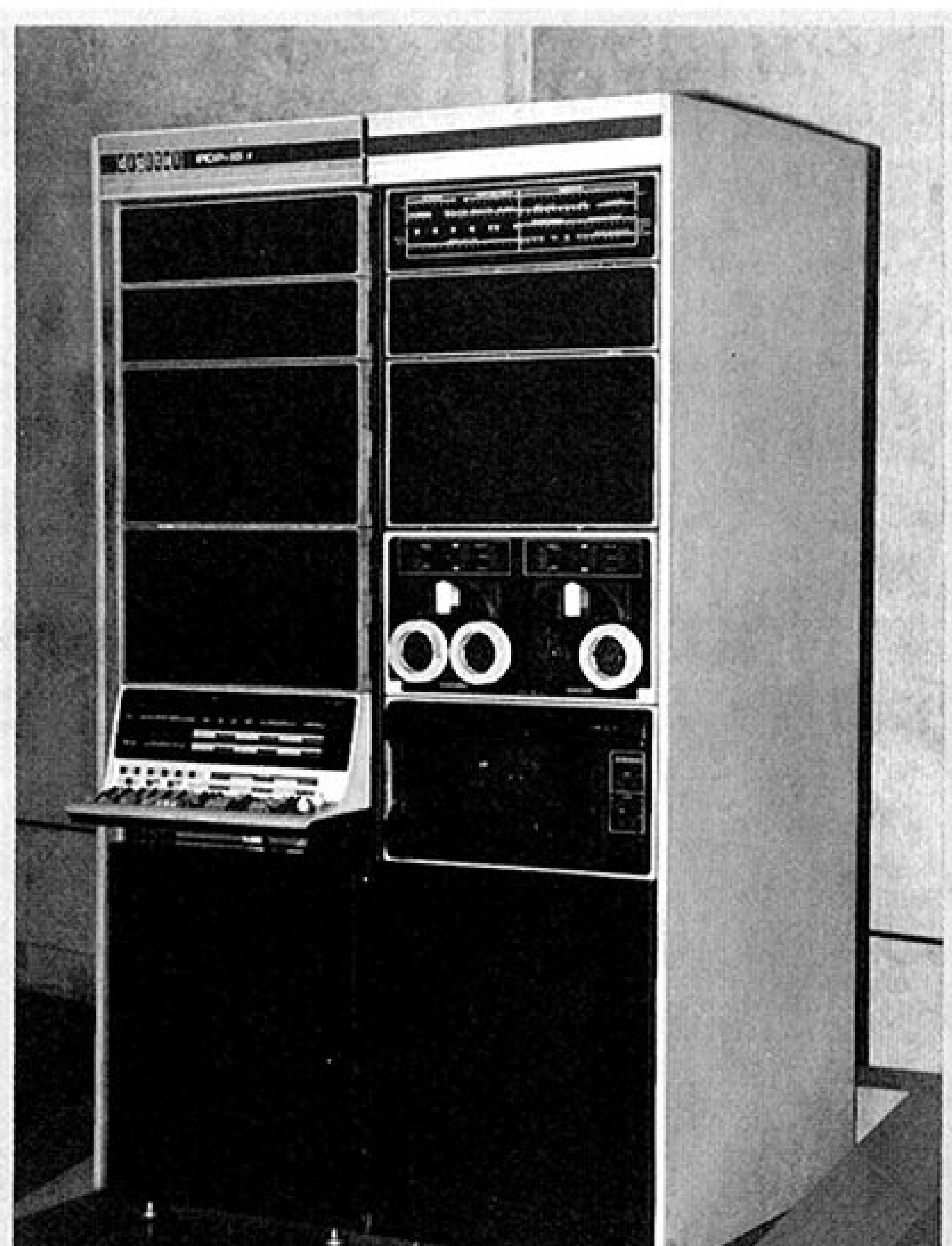


(a) PDP-8



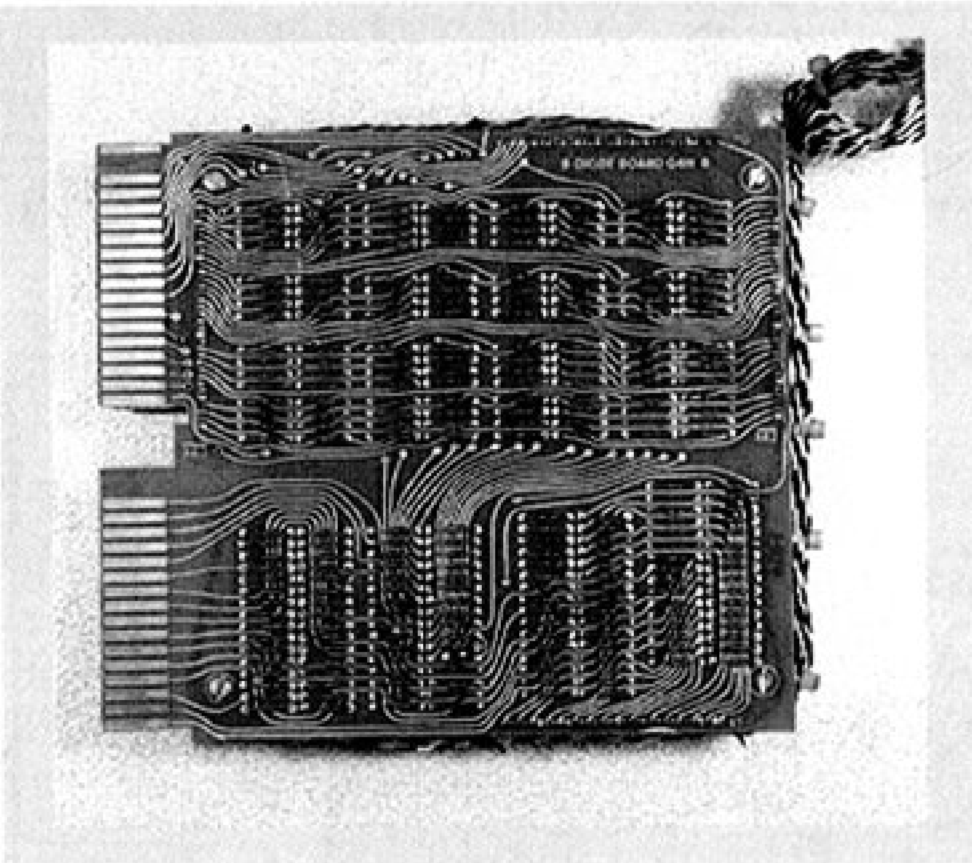
(b) PDP-15

〈写真 2〉 ディジタルモジュール

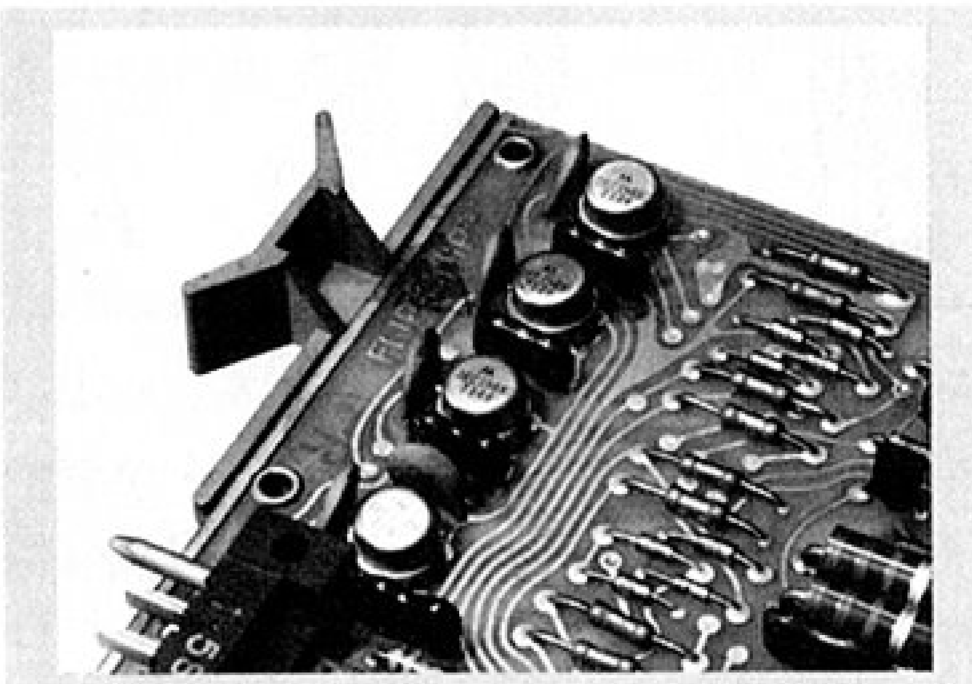


ワード長	18 ビット
最小メモリ	4 K ワード
最大メモリ	32 K ワード
キャッシュ	なし
サイクルタイム	1.0 μ 秒
アーキテクチャ	ワードマシン AC×1 I/O 命令あり
大きさ	高 1.8 m 幅 2.0 m 奥 0.6 m
重量(最小構成)	400 kg

〈写真 3〉 PDP-15 の外観と基本仕様



〈写真 4〉 コアメモリ基板

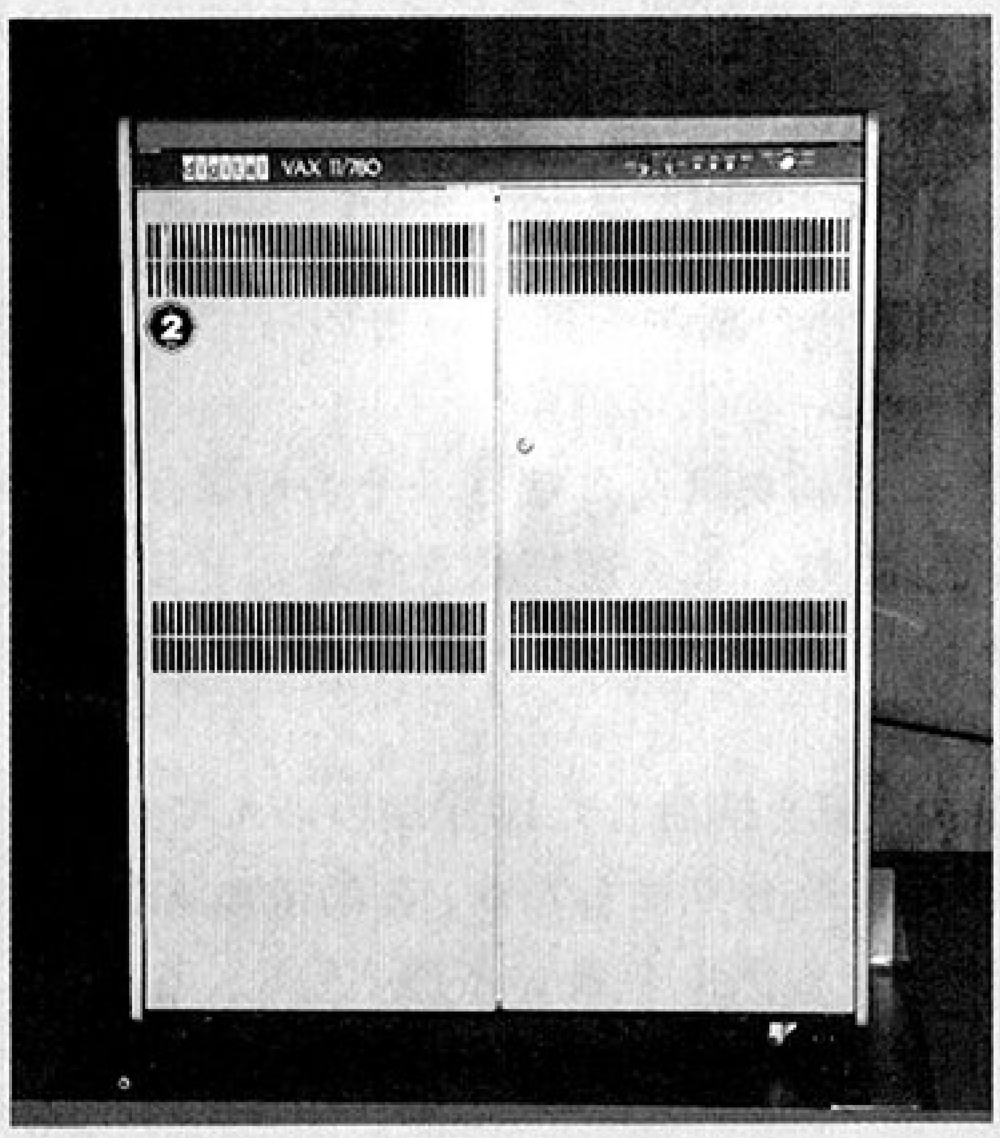


〈写真 5〉 コアメモリ制御基板の拡大

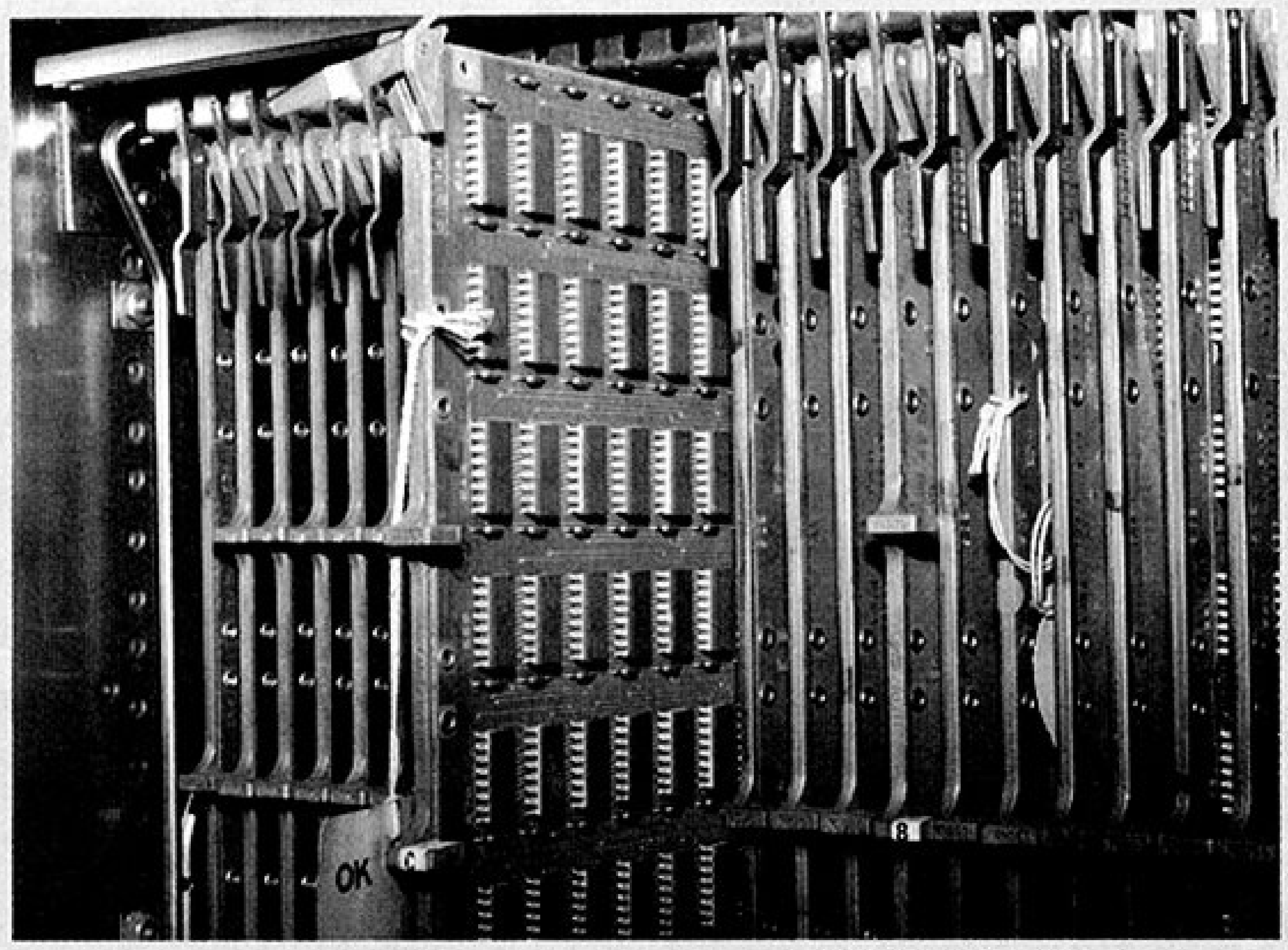


ワード長	16 ビット
最小メモリ	8 K バイト
最大メモリ	56 K バイト
キャッシュ	なし
サイクルタイム	0.8 μ 秒
アーキテクチャ	バイトマシン レジスタ 8 個 I/O 命令なし
大きさ	高 0.3 m 幅 0.5 m 奥 0.6 m
重量(最小構成)	40 kg

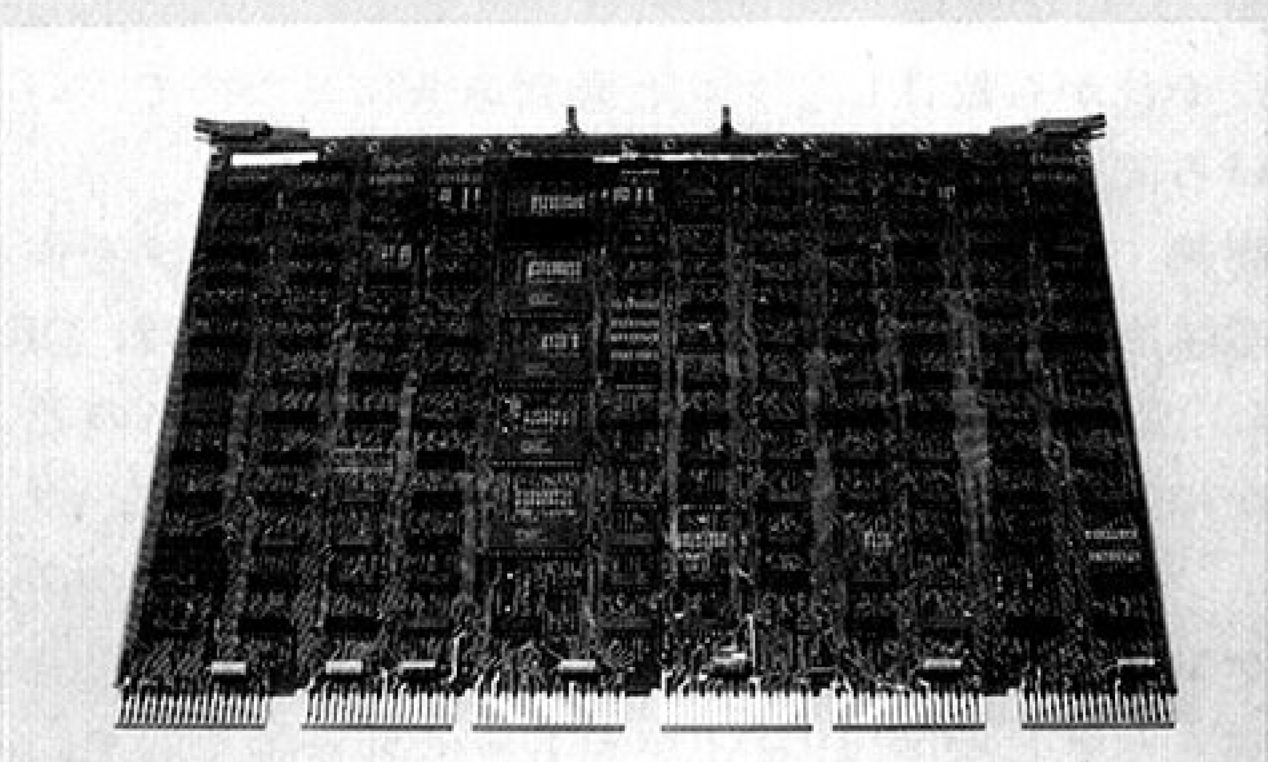
〈写真 6〉
PDP-11/20 の外観と
基本仕様



(a) VAX11/780 の外観



(b) VAX11/780 の CPU スロット(全 20 枚以上)



(c) VAX11/780 の CPU 基板

ワード長	32 ビット
最小メモリ	512 K バイト
最大メモリ	1 M バイト(初期の頃)
キャッシュ	8 K バイト
サイクルタイム	0.2 μ 秒
アーキテクチャ	可変長バイトマシン レジスタ 16 個 I/O 命令なし 仮想記憶方式
大きさ	高 1.6 m 幅 1.2 m 奥 0.8 m
重量(最小構成)	500 kg

〈写真 7〉 VAX11/780 の外観および CPU 基板

ミニコンピュータの夜明け

PDP-1からバイトマシンへの移行

● プロローグ

ここはどこなんだろう？ 古ぼけた壁，でたらめに繋ぎ合わされた床や階段．見覚えがあるなと思いながら先へ進んだ．



するとぽつんと明かりがついた部屋からメロディがなんともぎこちない音で聞こえてくる．

「Are you going to Scarborough Fair ...」

これはサイモンとガーファンクルの歌じゃないか．それにしても幻滅だな，もう少しうまくハミングしてくれよなと思いながら部屋を覗いた．

すると，かのLINC8(おもに研究所で使われたミニコン)が薄暗い部屋の中でオペレータパネルのランプを意味ありげに点滅させながら歌ってるじゃないか，と突然，中から大男が現れ，

「ウェルカム，アイビンルキンフォワードゥミーティンユー(あいたかったよ)」

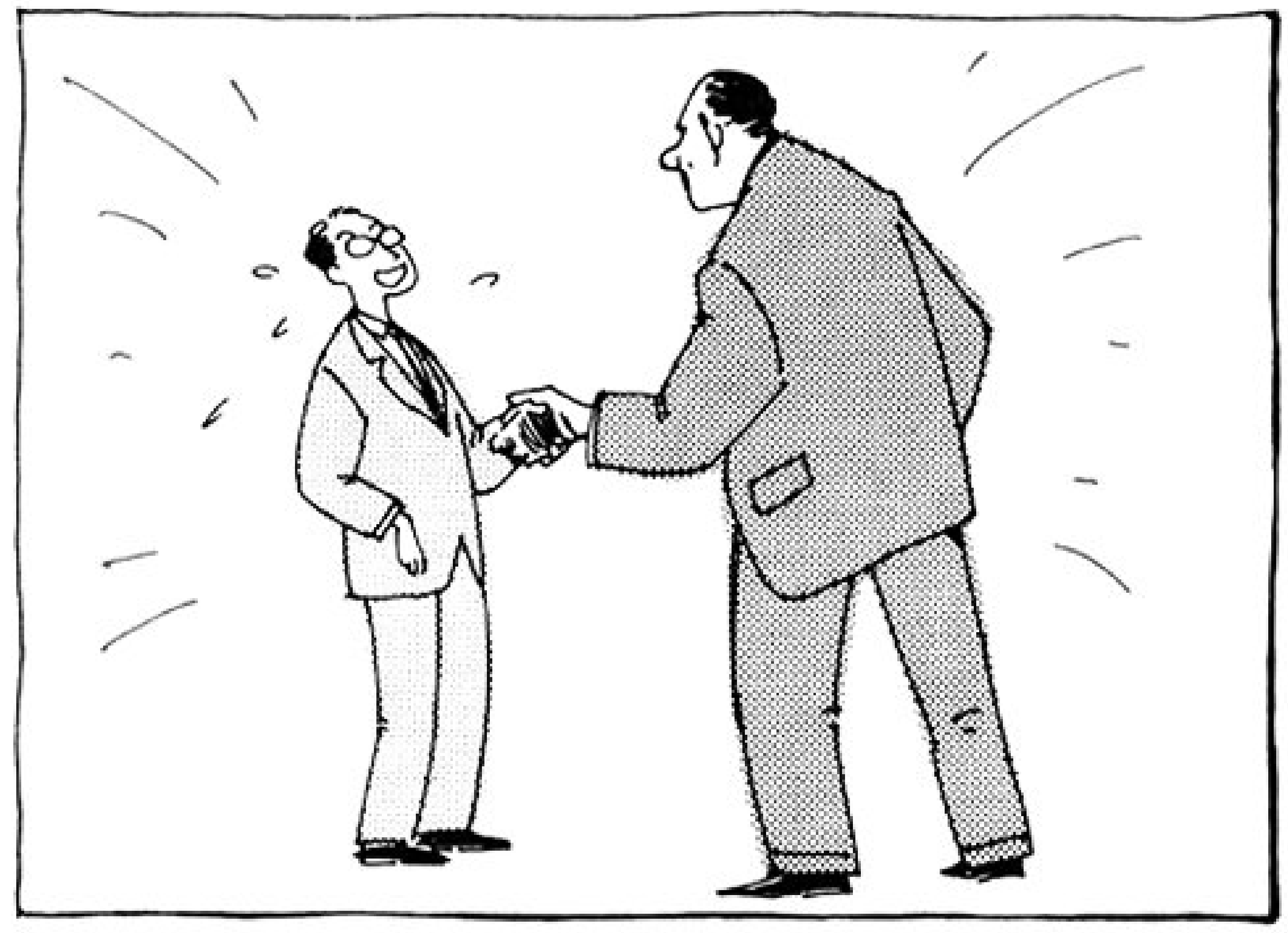
あまりの大きさと，毛むくじゃらな手がのびて握手を求めたので，なんだこれはスターウォーズのチューバッカかと驚いていると，

「DECの新しいコンピュータについてアーキテクチャはどうするか話をしたい」

これについては僕もよく知っているので，なんとか話ができそうだ．また，今まで疑問に思っていたことなどを聞き出すチャンスではないかと思い，つたない英語で，

「アイウッナッハヴミスジスゲットゥゲタフォーザワールドゥ，コーゥミ，カツオ(このような集まりに招いてもらい光栄だよ)」

そう，そこにいた人達はあのDEC伝説の人，ケン・オルセン(例の大男)，ゴードン・ベル，エドソン・デカストロなどのそうそうたるメンバーばかりではないか．



ケン・オルセン(以下ケン) 「DECは今非常な外圧にさらされているんだ．OEMメーカーが『IBMや他のメーカーはバイトマシンを出しているのに，DECは12ビットのPDP-8シリーズなどのワードマシンだけではないか，もっと高性能な，しかもバイトマシンを出してもらわないとDECのマシンでは商売ができん』と言ってきておるんだ」

勝男 「そうなんですか．なるほど．で，どのあたりまで話が進んでるんですか？」

ケン 「エドが自分の案が一番いいといって譲らないんだよ」

というような会話が延々と続き，その結果，エドの案は一時棚上げされ，他の可能性を探ることになった．

ケン 「我々も素晴らしいマシンを出さんとつぶれてしまう」

勝男 「僕も他の案を検討したほうがいいんでは」

と言ってこの場は終わりましたが，この後エドはDECを去りデータジェネラルを興し名機NOVAを出すことになろうとは，私(勝男)以外には知るよしもなかったのです．

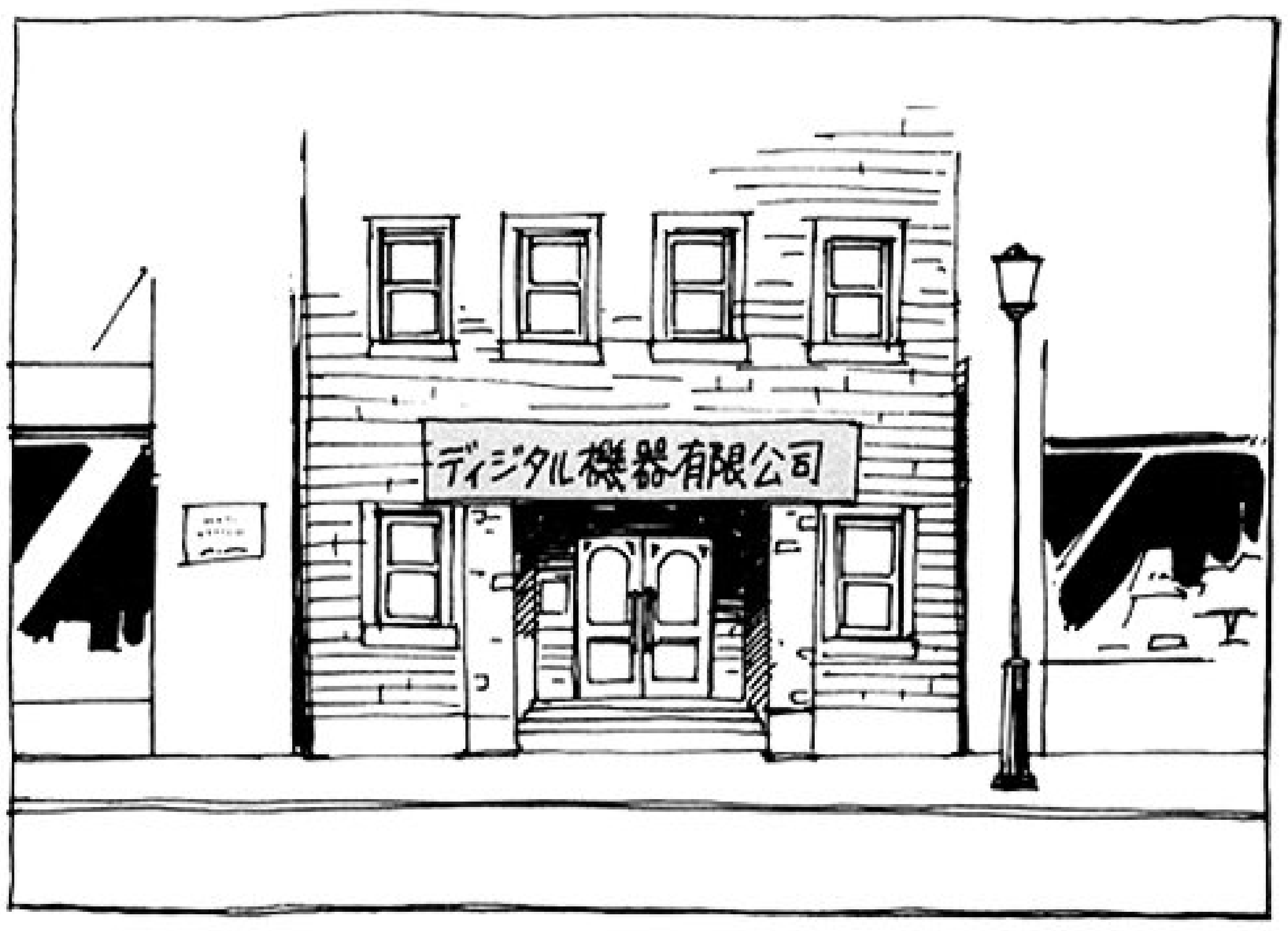
● DECの誕生

それから数日してケンと昼食を共にしたので，いろいろ聞いてみようと思った．

勝男 「ケン，あなたの理想とするコンピュータを作る意味あいからもコンピュータのコの字くらいは，DEC(Digital Equipment Corporation)という社名のどこかに入っていてもいいのではと思うんですが，なぜ『ディジタル(Digital)機器(Equipment)』なんですか？」

ケン 「勝男，よく聞いてくれたね．実は我々が会社を興そうとした頃はIBMがめちゃくちゃ強くって，それ以外のメーカーは青息吐息さ．そこでもってコンピュータなんかやるっていうとどこも金を出さないし，取り引きさえ危ないんだ．

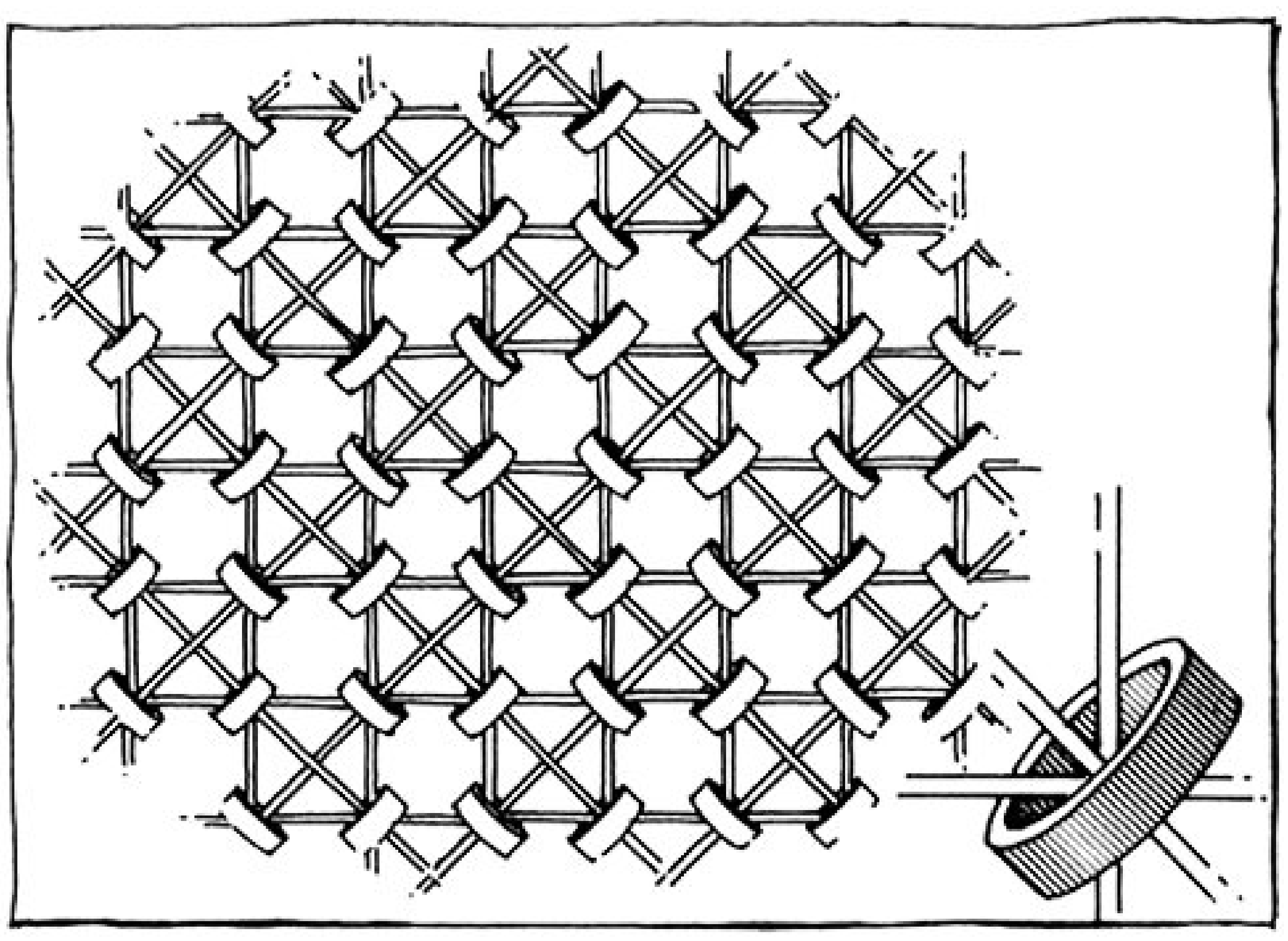
そこでね，最初はMITのWHIRLWINDプロジェクトで培った技術をもとに，ディジタル回路を基板に組み込んだ，『ディジタルモジュール』を作ることにしたんだ．それでコンピュータもディジタル系だし，じゃー，ディジタル機器でいこうということでDECになったんだよ」



勝男は，ケンにその頃の技術背景についても聞いてみた．

ケン　「そうだな，1952年にIBMが世界初の商用コンピュータ701なんてものを出し，そのメモリ素子が静電記憶管なんてものを使っていたんだ．それとほぼ同時期に私もかかわったWHIRLWINDの記憶装置としてコアメモリが開発されたんだよ．それは非常に小型で故障も少なく，それからしばらくコアメモリが全盛となったんだ．

コアメモリはドーナツのような形で，大きさは2～3ミリメートルくらいかな，我々がPDP-1を作った1960年には1ミリメートルくらいまで小さいのが出てきて，アクセススピードは2～3マイクロ秒だったな．いやー，素晴らしく速かったよ」



勝男はWHIRLWINDについて，どんなものか聞いてみた．

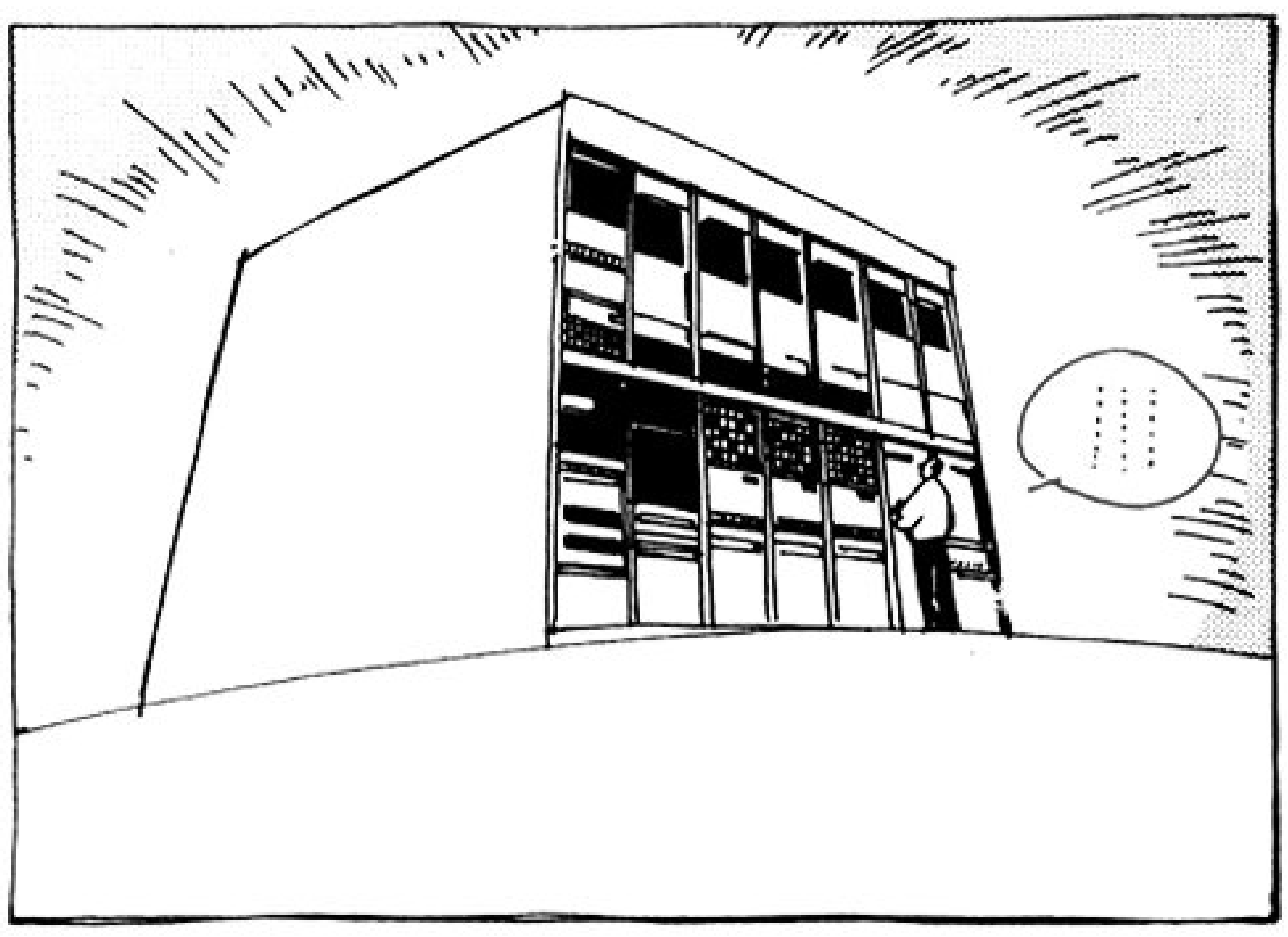
ケン　「WHIRLWINDは私も参加したコンピュータのプロジェクトでMIT(マサチューセッツ工科大学)で1945年に開発を始め，その名のとおり『旋風』をまき起こそうという意味あいがあったんだ．

真空管2800本を用いたプログラム内蔵式のプロトタイプが1950年に完成して，その後3年を経て真空管5000本，ダイオード約10000本，コアメモリを使用，1ワードは16ビット，そして大きさはENIACよりもずっと小型で縦横15メートル，高さ6メートルの完成品ができたんだ．

その時我々は，近い将来もっと小型のシステムが出現するのではという期待をもったんだ．これに使われたコアメモリのテスト用コンピュータMTC(メモリテスト・コンピュータ)の開発リーダーが私だったんだよ．

1950年代から60年代にかけては，IBMに代表されるようにコンピュータはどんどん大型になるし，また空調の施された専用の巨大なコンピュータ室に設置して使用されて，CPU，記憶装置，そして入出力を専門に行うミニコンピュータ以上の大きさと性能を持つチャネルと呼ばれる装置も必要だったんだ．

価格はなんと100万ドル以上もしたんだ．でも私はそんなあり方に疑問をもって，『コンピュータはもっと身近な道具でなければならない』と思い，わずか4名でDECを作ったのさ」



勝男　「ケン，100万ドルって約1億円じゃないか．大型だからそんなに高いとは思わないけど」

ケン　「何言ってるんだよ勝男，そのころは1ドル＝360円だから3億6千万円じゃないか」

勝男　「あーそうだった勘違いしてたよ」

ケン　「1960年になると，ディジタルモジュールで身につけた技術をもとに，PDP-1という世界で最初の18ビットのミニコンピュータを出したんだ」

勝男　「ケン，疑問に思ってたんだがPDPってどういう意味？ それに，なぜ18ビットなの？ なぜ8の倍数ではなく6の倍数を基準としたミニコンでなければならなかったの？」

ケン　「よく聞いてくれたね．まず，PDPとはProgrammed Data Processorのことさ，読んで字のごとく．次の質問だが，その当時必要とされていたデータ

は6ビットでいろいろな文字コードや制御コードを表現していたんだよ．標準規格で名高いISO, JISのように将来の拡張性などを考慮して8ビットコードが定義されたのは後のことで，実験機器，A-Dコンバータなどは精度が低く6ビットもあればよかったんだ．

それで，3ビットの整数倍のほうが扱いが簡単だということで，DECでは8進数を用いようと決めたんだ．勝男は数字を数える時に16進で0，1，2，…9，A, B, C, D, E, Fなんてやるか？ デバッグする時なんて面倒なだけだよ．例えば，**1F35**と**AFDE**の足し算なんて考えただけでもぞっとするね．

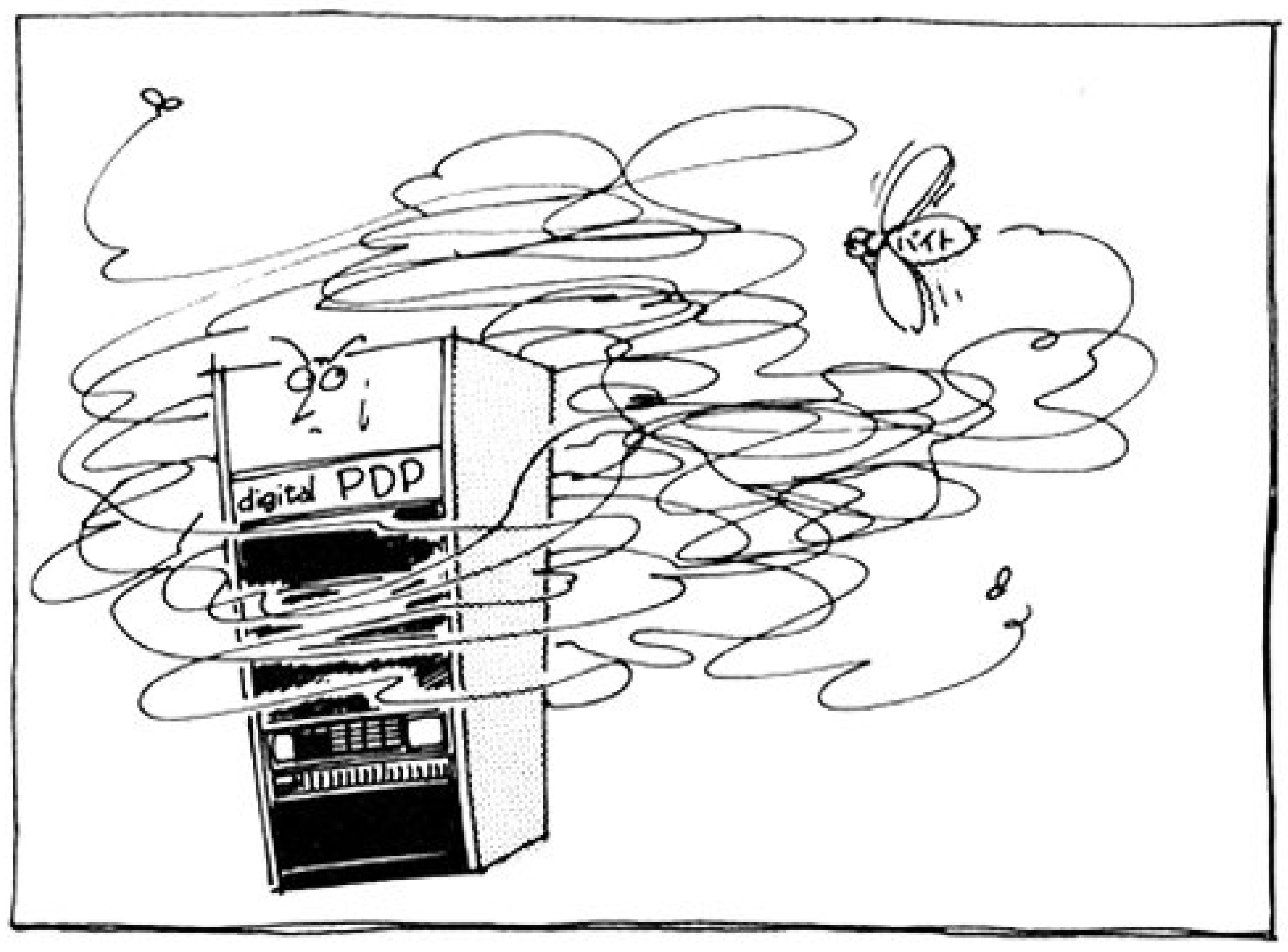
それにデータを保存するためのオープンリール・タイプの磁気テープも6ビット単位で記録したものがあったんだ．それで1ワードに6の整数倍のデータを保存できれば効率がいいじゃないかということなんだよ．

それにMITが当時TX0というコンピュータの開発を行い，そのプロジェクトで最初に開発されたコンピュータが18ビットだったんだ．そこで，それを手本としてDECではまず18ビットのコンピュータを製造したのさ．

一度作ってユーザーがいいと思えば6の倍数でも不便を感じなくなっていて，DECのコンピュータは大学や研究所に受け入れられたんだ．

そして，たくさんのソフトウェアがユーザーにより開発されたんだ．1961年にユーザーの集まりであるDECUS(Digital Equipment User's Society)が発足したんだよ．

しかしだな，最近はバイト，バイトってうるさくて困ってるよ．」



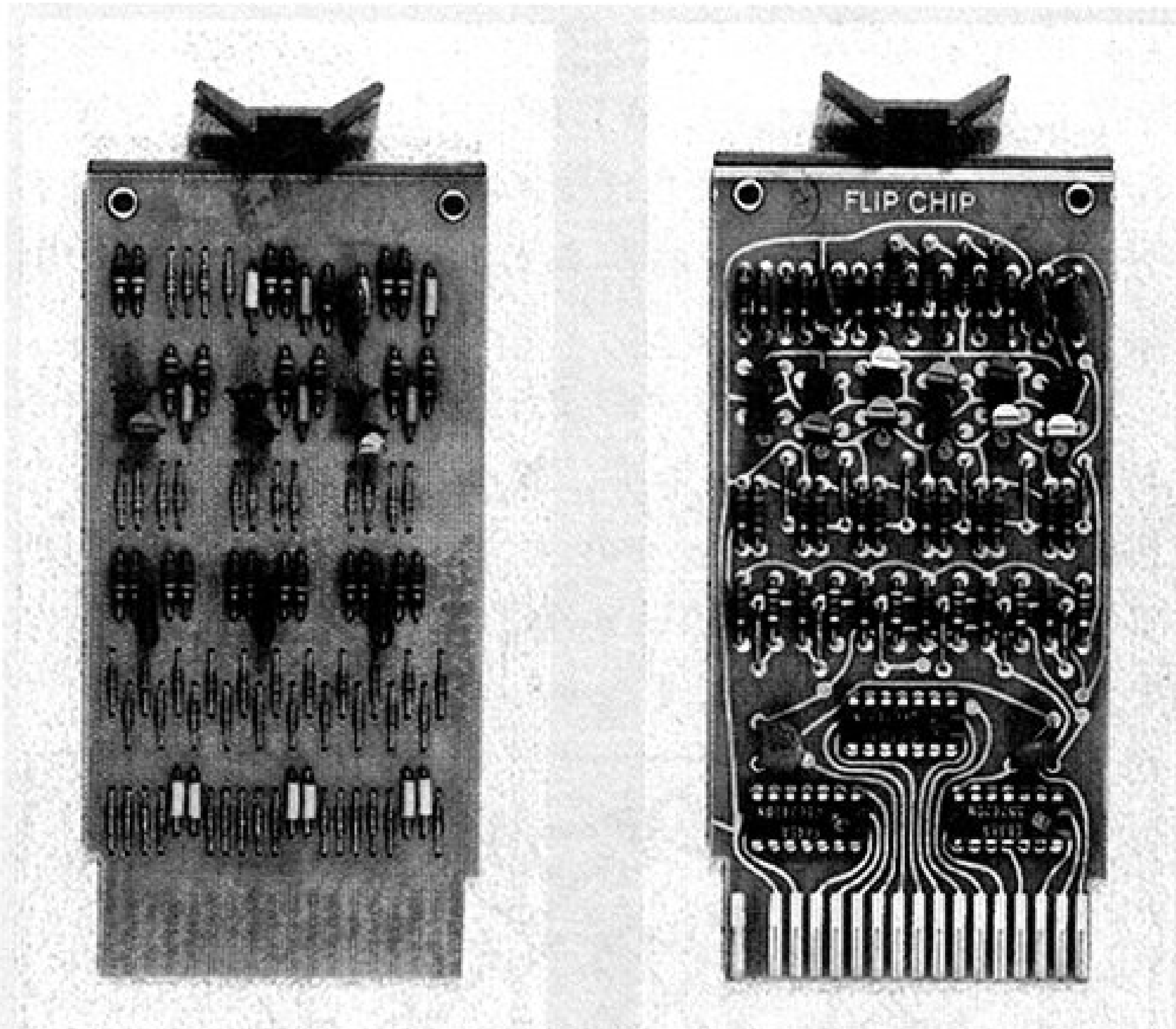
〈表1〉PDPシリーズの変遷

語長	システム名
12ビット	PDP-5(1963年)，リアルタイム，制御用として使用された
	PDP-8(1965年)，スピード1.5マイクロセコンド，4Kワードコアメモリ(最大32Kワード)，1.8万ドル，DECの礎を築いた．このシステムからミニコンと呼ばれるようになった
	LINC8(Laboratory Instrument Computer)(1966年)は1962年MITで開発されたLINCをモデルとして開発され，生物や工学の分野で利用された． A-Dコンバータやスピーカ，コンパクトなOSを装備し，簡単に実験データを取り込めた．デモでサイモン&ガーファンクルなどいろいろなジャンルの音楽を奏でた． このシステムは，PDP-8とLINCシステムとを単純に一つのシステムにまとめ，命令によりどちらのシステムモードを使用するかを切り替えた
	PDP-12(1969年)，LINC8をIC技術を用いて効果的にかつコンパクトに仕上げたシステム
	PDP-8E(1970年)，それまでの8シリーズをIC，LSIを用いて非常にコンパクトにまとめ上げたシステム．OMNIBUSというどのスロットにモジュールを入れても動作する特色あるバスを採用した
18ビット	PDP-1(1960年)，スピード5マイクロセコンド，4Kワードコアメモリ，タイプライター，ペーパーテープ，12万ドル
	PDP-4(1963年)，PDP-7(1964年)
	PDP-9(1966年)，PDP-15(1969年)，CADで有名
36ビット	PDP-6(1964年)，0.25 MIPS，約12～30万ドル
	PDP-10(1971年)，大型のシステムで約2億円ほどした．その後DECsystem10/20となり，世界中の人工知能研究の分野，第五世代コンピュータ開発などで用いられた

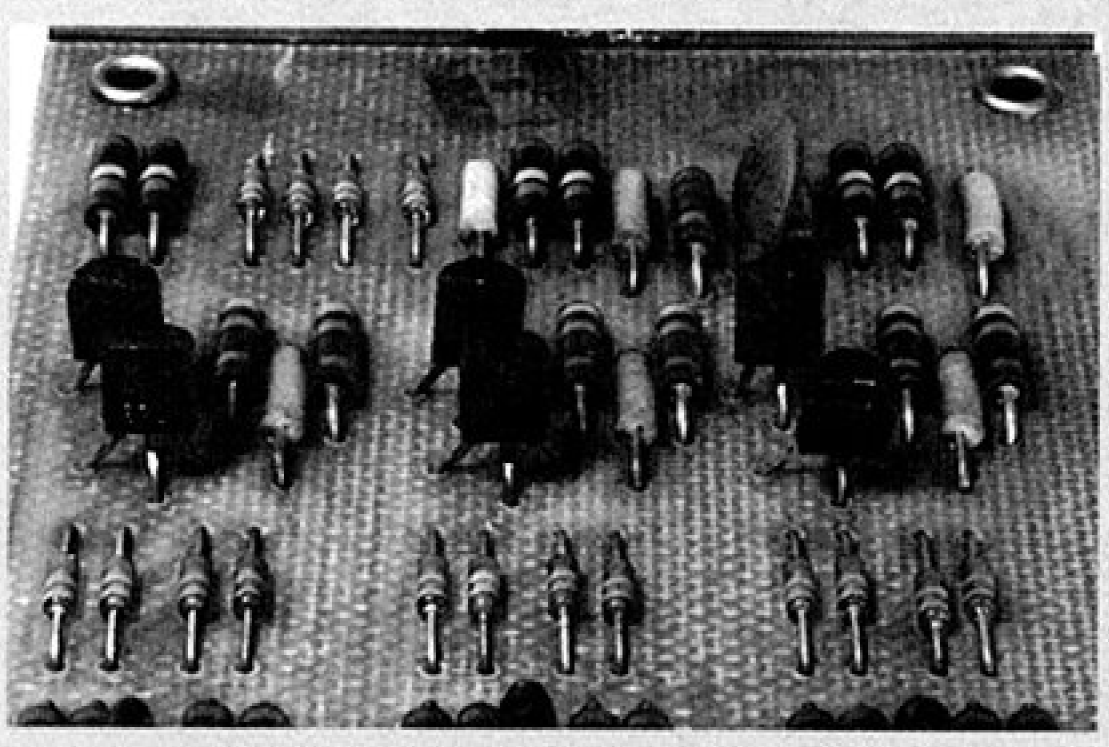
● DECのミニコンの系譜

勝男「じゃー，ケン，今までどんなPDPを作ってきたの？ 系統だてて教えてくれない？」

ケン「それは簡単だよ，私が今まで作ってきたコンピュータを表にしてまとめてあるから，それを見よう(**表1**)」

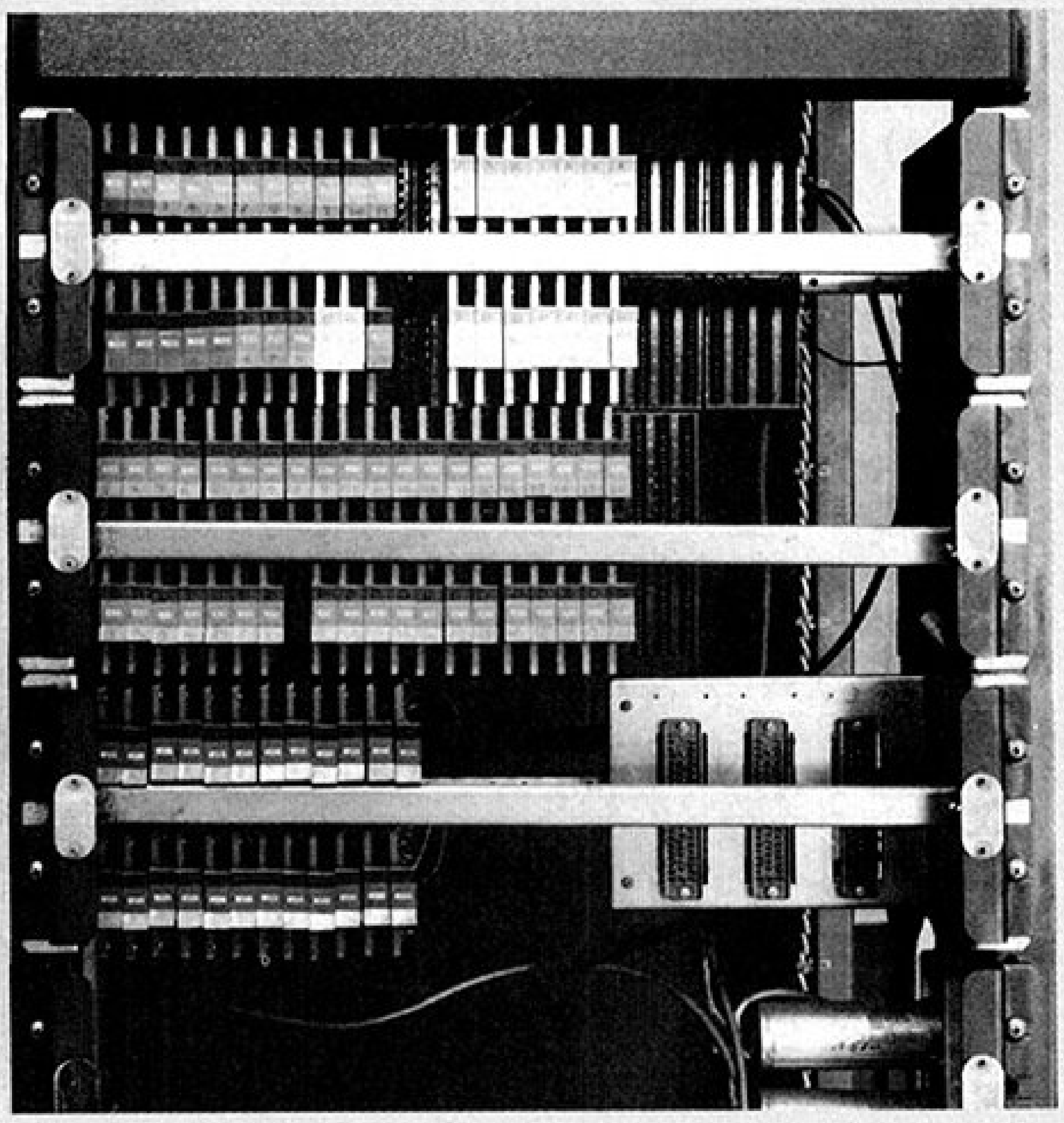


(a) ディジタルモジュールの外観

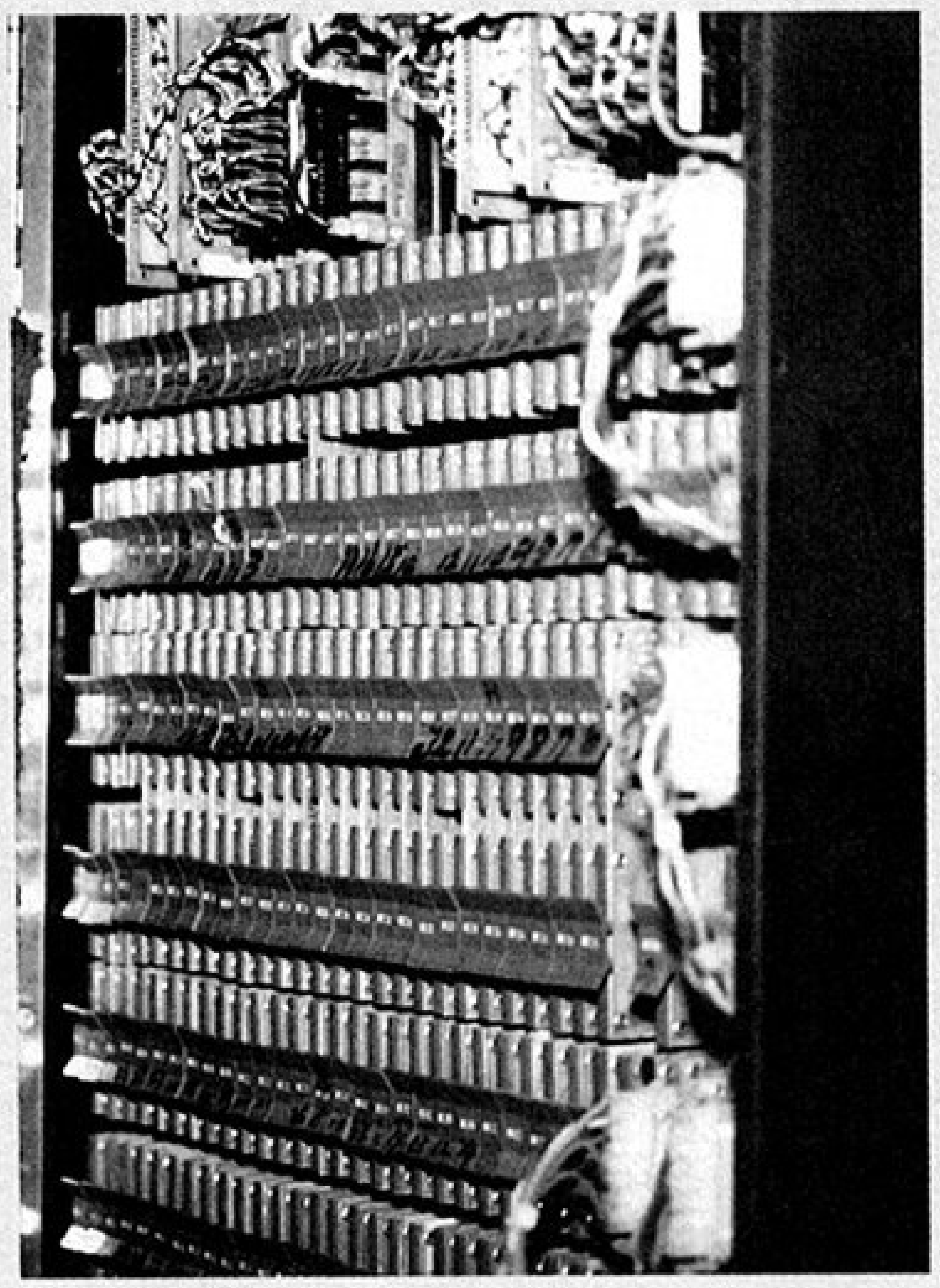


(b) ディジタルモジュールの拡大

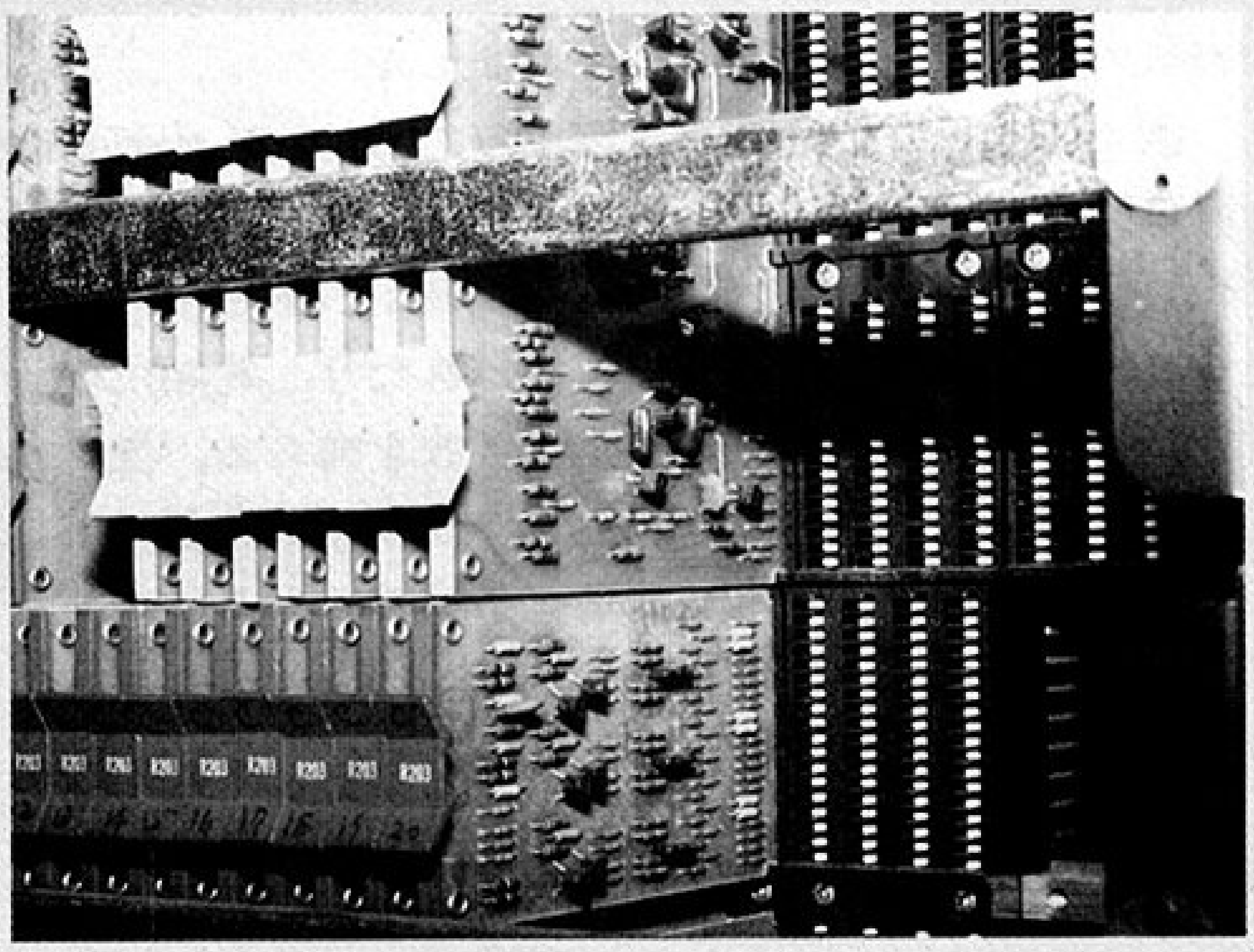
〈写真1〉ディジタルモジュール



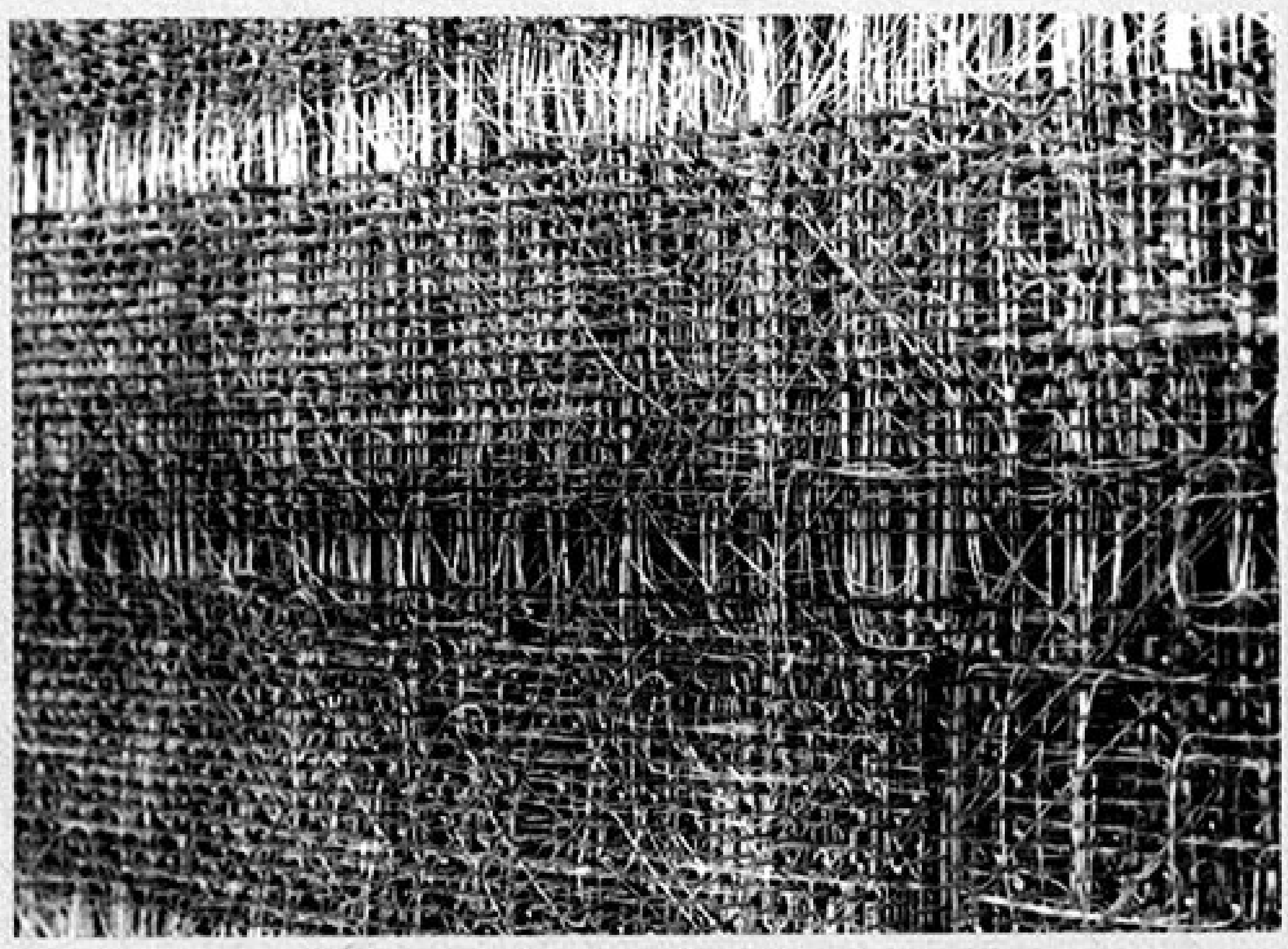
(a) PDP-8のスロット



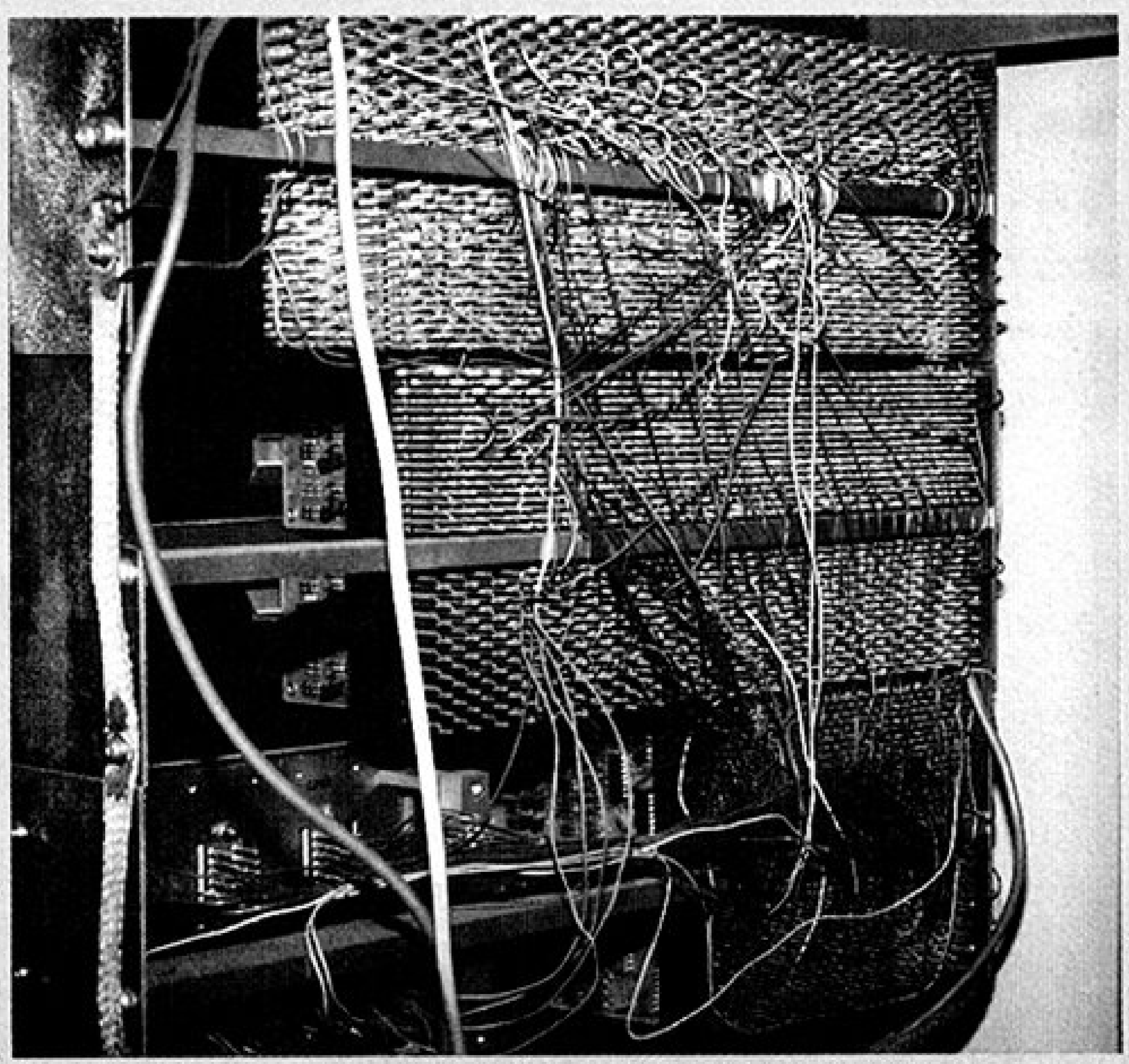
(d) PDP-15のスロット．PDP-15では，ディジタルモジュールを内側から装着する



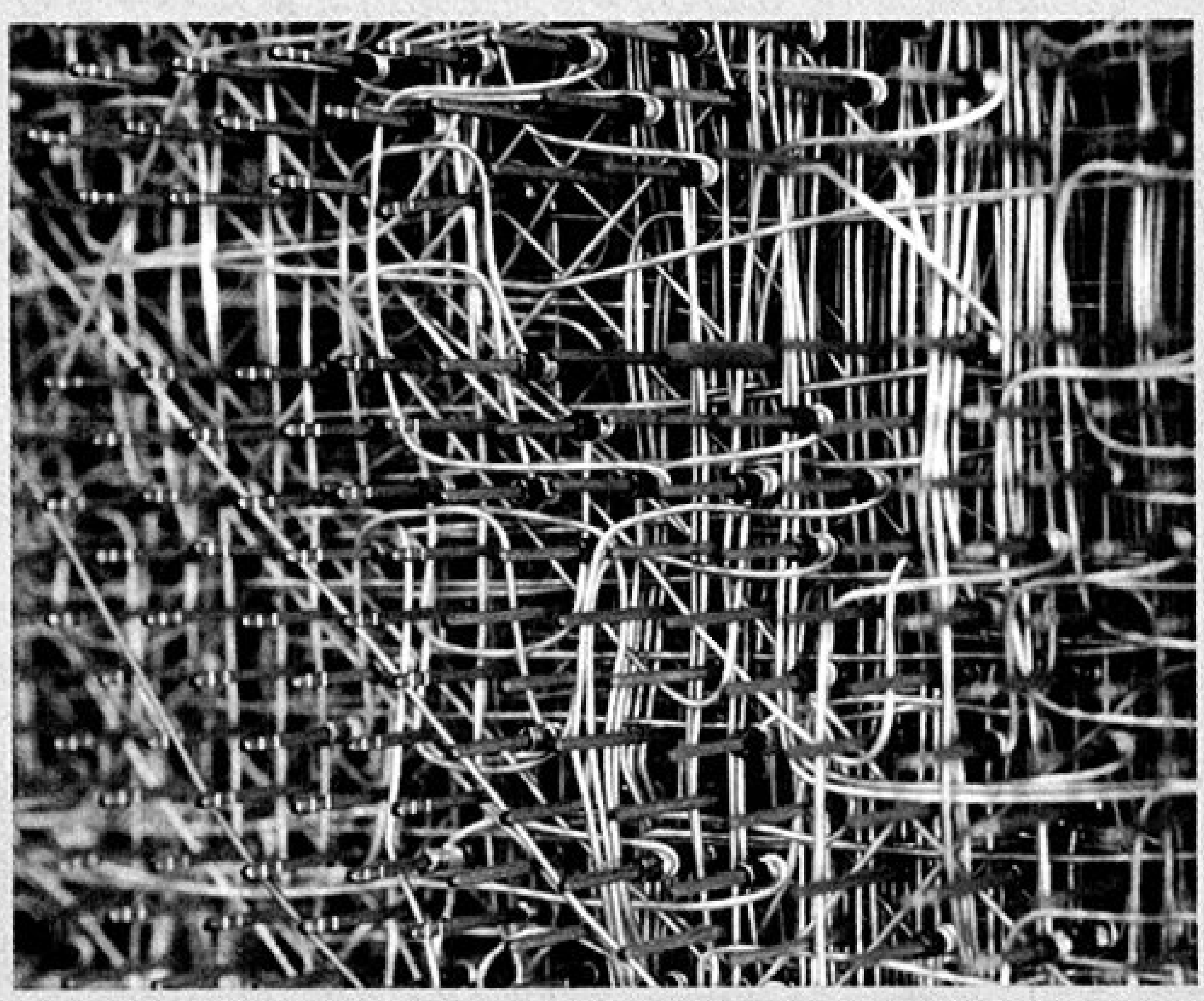
(b) スロットとディジタルモジュールの接合部



(e) PDP-15のバックプレーン．(d)の反対側



(c) バックプレーンのようす．(b)の反対側



(f) (e)の拡大写真．1本1本線が巻き付けられている

〈写真2〉 バックプレーン

勝男　「ケン，PDP-2，PDP-3 が抜けてるじゃないか」

ケン　「それは開発したんだけど製品化されなかったんだよ．PDP-2 は 24 ビット，PDP-3 は 36 ビットだった」

勝男　「当時のコンピュータはどんな手順で設計/製造されたの？」

ケン　「そうだね，PDP-1 の頃は縦 20 センチ，横 10 センチのボードにトランジスタ，ダイオード，抵抗，キャパシタ(コンデンサ)などを組み合わせて，なんと 1 個か 2 個のフリップフロップや AND/OR/NOT 回路などを構成したんだよ(**写真 1**)．

そして，これらのボードを数百枚から数千枚組み合わせてバックプレーンと呼ばれるスロットに差し込み，その背面部分を人間が配線図を基にして細い線でつなぎシステムを作ったんだ(**写真 2**)．

ところで勝男は論理値の 0/1 は 0/3(ボルト)に対応すると教わったと思うが，これら初期の回路はネガティブロジックとよばれる方式を採用していて，0 ボルトとマイナス 3 ボルトで論理値の 0，1 を代表させていたんだ．

ミニコンはユーザーがデバイスを簡単に接続できるという特徴をもっていて，初期の頃から入出力用にインターフェースを提供していたんだよ．

すなわちその頃は，システムの心臓部であるメモリアドレス，メモリデータ，I/O タイミングなどの信号をユーザーに開放したんだ」

勝男　「大胆だなあ．これではユーザーが設計ミスをしたらシステムまでも壊れてしまうじゃないか」

ケン　「すべてはユーザーのおぼしめしだよ(笑)」

ミニコンピュータへの道

● 大型でなくてもこんなことができるんだ!!

それからというものは，ケンたちは非常にハードなスケジュールをこなし，新しいバイトマシンのアーキテクチャをどうするかということに全力を注いだ．

ある日ケンが，PDP-8 に関しての技術セミナーの依頼をトラワーザという会社から受けてるんだが，時間の空いた時にやってくれないかと言ってきた．

相手は PDP-8 をこれから全社的に採用しようとしてるんでやさしく説明して欲しいとのこと．

これについてはよく知っていたのでひとつ返事で OK と答えましたが，いざ当日になると緊張で声が震え，

勝男　「ウ，ウ，ウレディースアンジェントゥーメン，アイム アツシ カツオ フロムジャパン」

顧客の一人　「おいおい緊張しすぎだぞ，LADY じゃなく READY に聞こえてるよ」

とそんなに発音悪かったっけと気を取り直し，それから顧客の自己紹介が始まった．その名前は非常にアメリカ人ばなれしていた．

非常な切れ者で，将来は自分でコンピュータを作るんだと言っていたイブム・アムダン．彼ならやるだろうと思った．

イタリア移民で，どっしりとして重厚そうな感じの大男イターキ・イターク．

独立心が旺盛で我が道をいく感じで，変わった名前のニキデ・ン・ニーク，しかしとってもビジネスがうまいという印象をもった．

そして，最後に精悍そのものでなんでも取り込んでしまいそうな感じがしたフュージット・フェイカム．

しかし僕は，イブム・アムダン以外の 3 人に対しては何となく身近な存在に思えたが，その理由が明らかになるのはまだずっと後のことだった．

● PDP-8 の出現

勝男　「PDP-8 は 1965 年に発売され，その簡単な命令体系と入出力により，いろいろなメーカーが採用して自社の装置に組み込んで使用しました．

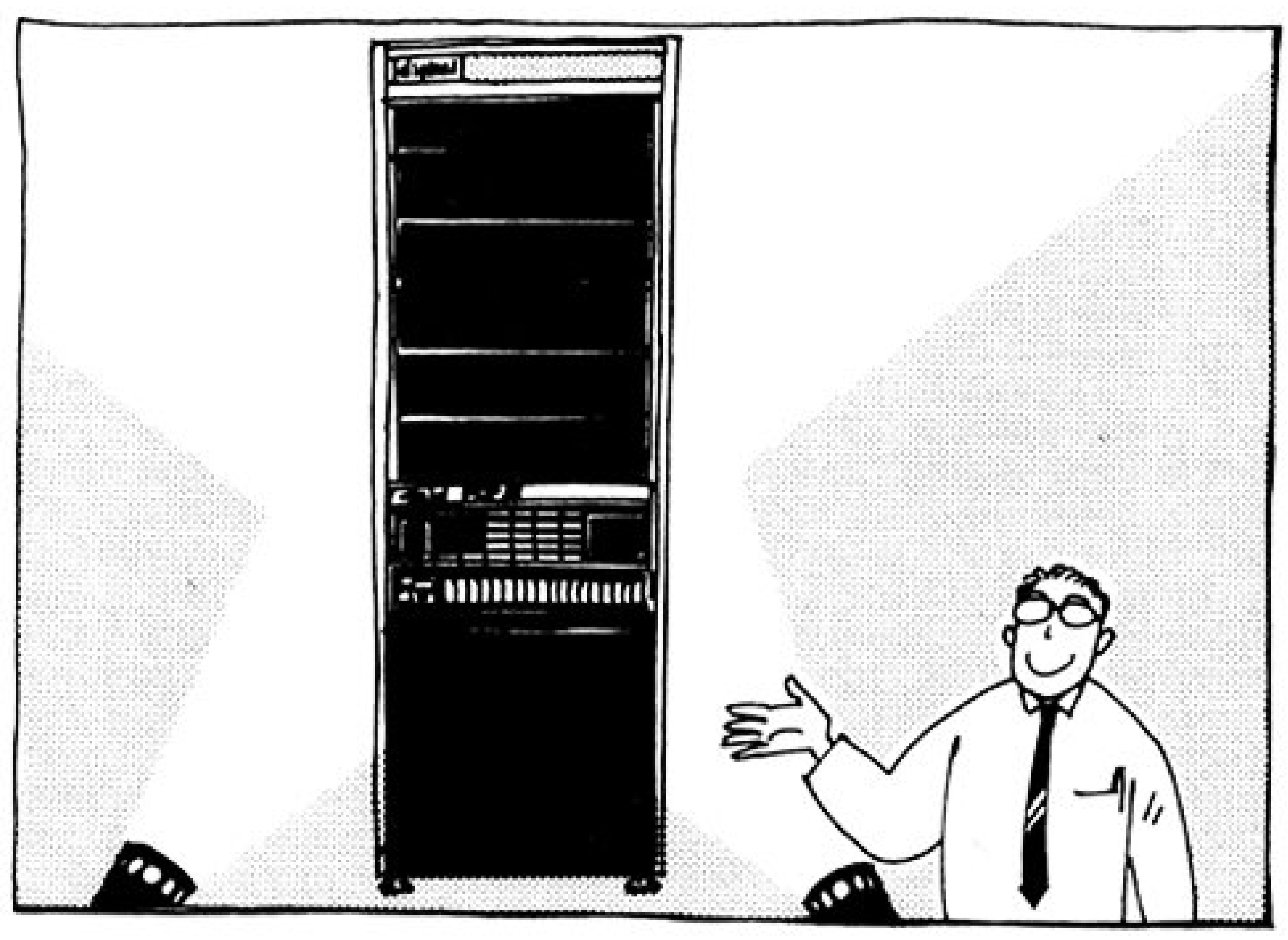
例えば，物質の構造を研究するために，X 線解析装置に接続し，そのデータを基に物質構造を決定するための演算をなんと 10 日間連続運転して答えを出したり，大量の血液の成分分析に使われたり，船舶に搭載して人工衛星から送られてくる電波を捕らえ，現在はやりの GPS の先駆けとなる，ドップラ効果を利用してその位置を示してくれるシステムや，MIDI のはしりとなる音楽用のアプリケーション，人間や猿や猫の脳の研究などにも用いられました．それはなんと 20 数年も昔のことです」

イブム・アムダン　「GPS だの MIDI だの脳だの，挙げ句のはては 20 年も前だと，一体なんのことをいっとるんだ!!」

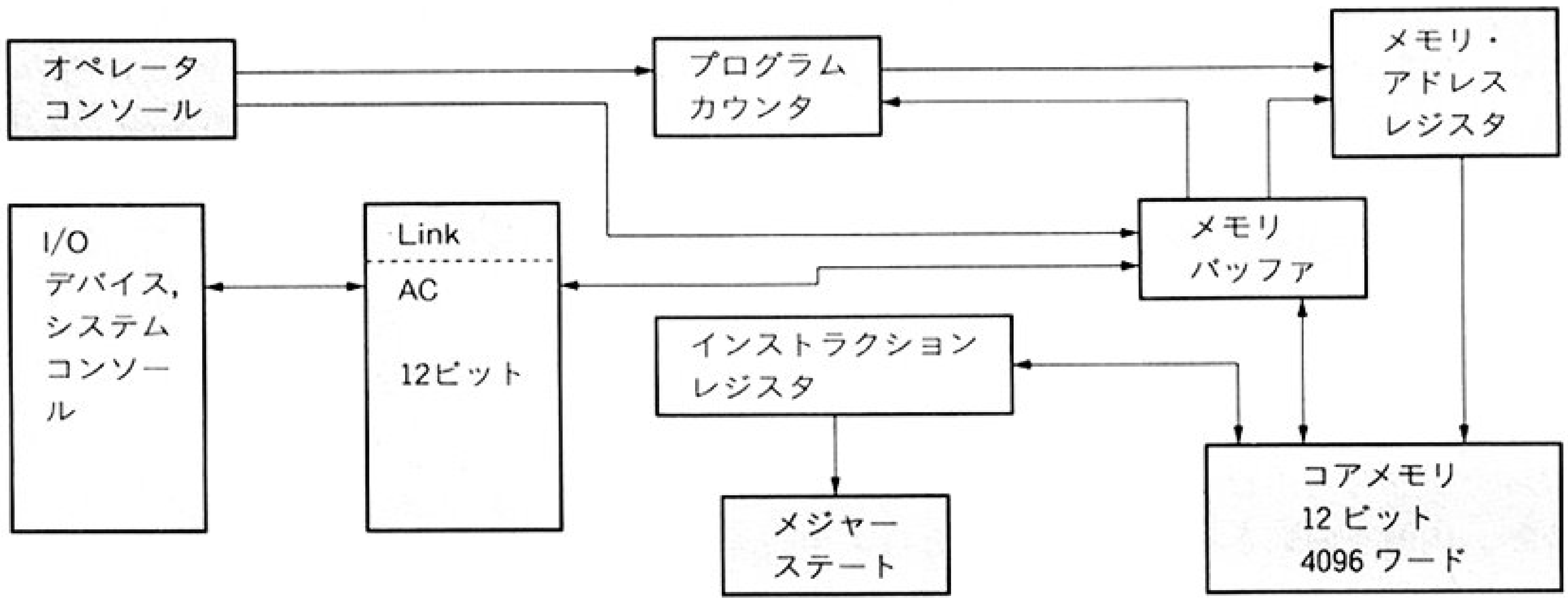
（そうだ今は 1970 年なんだ．）

勝男　「オーまだ動転してる」

と言い訳してから PDP-8 の構造について説明を始めました．



〈図1〉
PDP-8のブロック図



オペレータコンソール(写真4)	メモリに対してデータの書き込みや読み出しを行ったり，プログラムの起動や停止を行う
I/Oデバイス，システムコンソール	磁気ディスク，磁気テープ，A-Dコンバータ，紙テープリーダ/パンチャなどを接続する
プログラムカウンタ	命令の実行順序を制御する
Link/AC	Linkビットは AC(データの一時保管)と循環してつながっている
インストラクションレジスタ	命令を保持し解読などのために利用する
メジャーステート	CPUの三つある状態を表す(命令の取り込み，解読，実行)
メモリ・アドレスレジスタ	メモリのどの番地を指すか保持するためのもの
メモリバッファ	メモリに対する書き込み/読み出しデータを保持する
コアメモリ	プログラム/データを記憶しておく

● DEC最初のベストセラーPDP-8，主記憶は4Kワード

勝男　「PDP-8は12ビットのアキュムレータ1個，1ビットのLinkビット，そして標準構成で4Kワードの主記憶から構成されます．メモリパリティはありません．

最小構成で18000ドルですから皆さんは安いとお思いでしょう．また，コアメモリも4Kワードで，たったの6000ドルたらずです．

それに引き替え，大型のIBMや他のメーカーのは100万ドル以上もして，とても大きく，メモリも食うし，専用の冷房のよくきいたコンピュータ室が必要で，簡単に人が扱えるものじゃありません．

話を戻して，コアメモリは虫眼鏡でみるとドーナツ状の形(数分の1ミリ)をしていて，これにアドレス選択線2本，データ書き込み/読み出し線1本を通した構造になっています．

これが12ビット×1024×4個(＝4Kワード)を単位として構成されています．コアメモリの製造には僕の故郷日本の織物の技術が用いられていて，コアに線を通して，整然と配列させるために利用されました．このタイプのメモリは，電源が落ちてもその内容は半永久的に保持されます．

実際に，3年間まったく使われていなかったPDP-8の電源を入れると問題なくプログラムが動作したという例があります．

現在よく用いられているDRAMは電源が落ちると内容が失われてしまい，内容を保持するためにはバッテリバックアップという装置が必要だということはご存知だと思います．

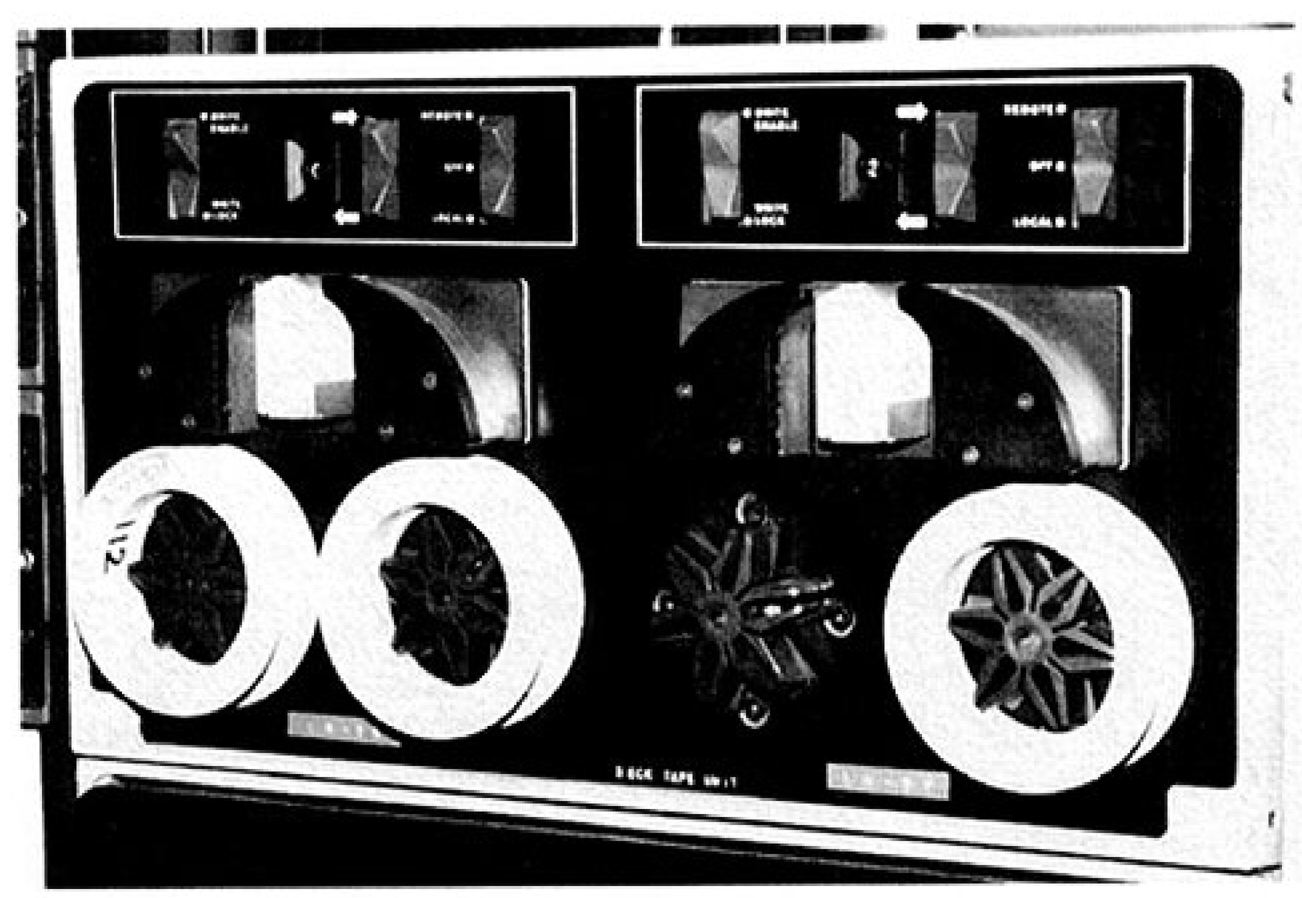
PDP-8はメモリを4Kワード以上に拡張する場合，メモリマネジメントと呼ばれるモジュールを追加する必要があります．

この機構ではインストラクションフィールド(IF)，データフィールド(DF)があり，プログラムとデータを分離して扱うことができます．では，PDP-8のブロックダイアグラムを図1に示しましょう．」

イブム・アムダン　「何だね，勝男，そのDRAMとかバッテリーバックアップというのは？」

(あっ，まだなかったっけと思いつつ，)

勝男　「エー，こんなメモリがあればな～と，僕の希望です」



〈写真3〉 DECtape(PDP-15)

● PDP-8のソフトウェア

勝男 「PDP-8では，ユーザーがアプリケーションを開発する場合はアセンブラを用いますが，開発のためにオペレーティングシステムが必要と感じた場合には磁気テープ(DECtape)(**写真3**)か磁気ディスク上で動作するオペレーティングシステムを購入します．

磁気ディスクは固定ヘッドタイプと呼ばれ，まさに固定された数十個のヘッドの上に直径30センチの磁性体を塗布した円盤を取り付けて，ヘッドと接触させながら円盤を回転させ，データの書き込みや読み出しを行います．1セクターは1ワードで構成され，何と32Kワードの大容量です．

また，テープにはタイミングやアドレスなどのマークが付けられています．

このDECtape上で走るOSもあり，FORTRANなどのコンパイラやアプリケーションを実行できます．

PDP-8のオペレータコンソールは，非常に多くのスイッチやランプ類で構成されています(**写真4**)．スイッチの機能は，前に簡単に述べたようにメモリに対するアクセスやシステムの停止，スタートを行います．

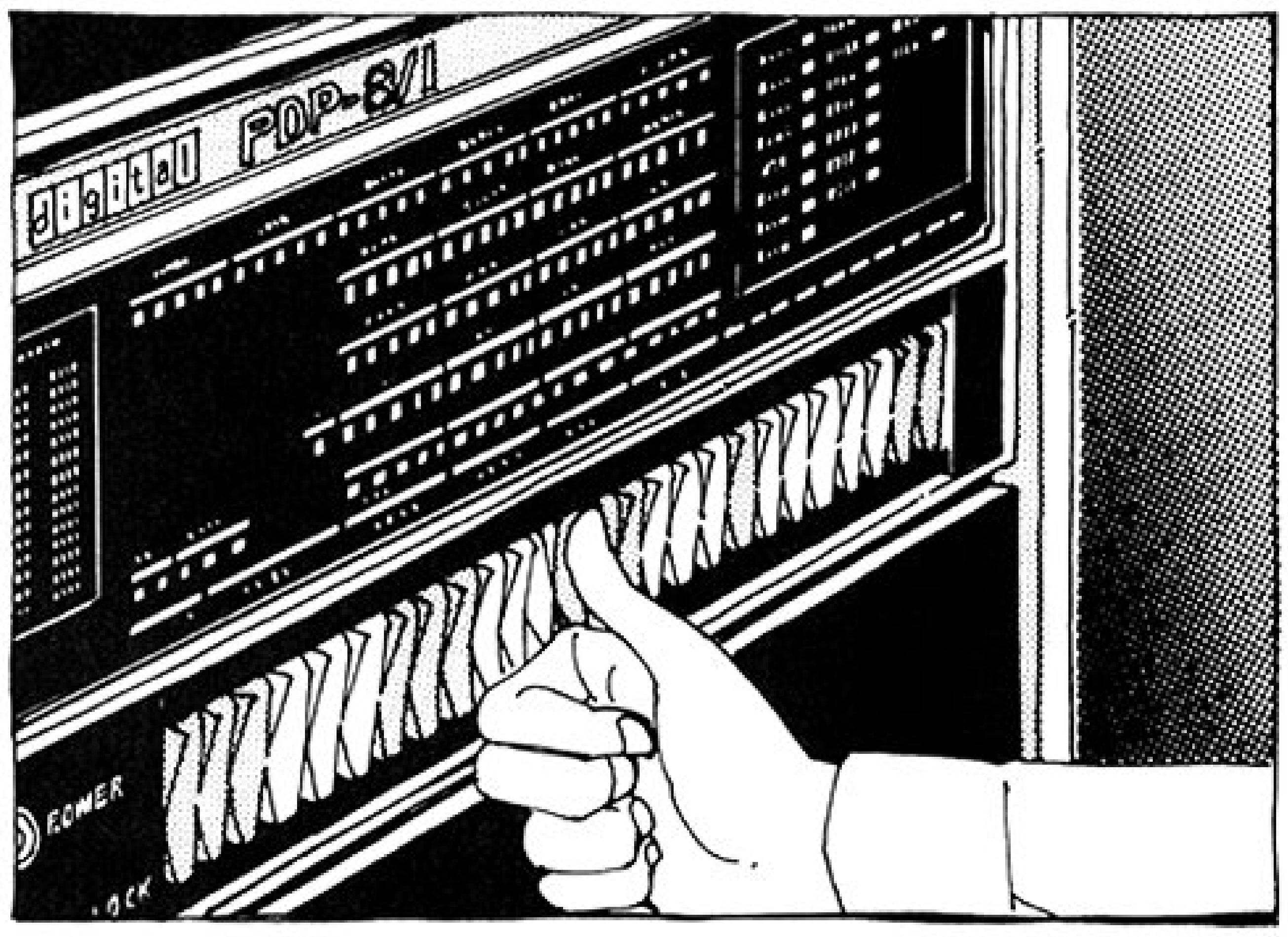
ランプは約100個近くあり，システムの重要な信号を表示していて，故障時にはこれを頼りに保守を行います．大型のシステムでは，なんとこのような信号を数千ヶ所も見ることができます．

オペレータコンソールのよいところは，ランプが点滅するパターンで自分の作ったプログラムがどのあたりまで実行されたかの見当もつけられるところがいいですね．では質問がなければ次にいきます」

とすかさず，

イブム・アムダン 「DECtape上で…非常に高速というがどのくらいか？」

勝男 「テープの端から端までのアクセス時間はわずか30秒です」



イブム・アムダン 「そんなに速いのか．信じられん」という顔を見てたいしたもんだミニコンはと思った．

勝男 「PDP-8のシステムコンソールはテレタイプ(タイプライタに似ている)と呼ばれるもので，1秒間に10文字の印字やキー入力，紙テープの読み取り，紙テープのパンチができる能力をもっています．

しかもすべて機械式で信頼性抜群，音が比較的静かです．コンピュータとのつなぎの所で，カーボンの棒をローターに装着した構造によりパラレル/シリアル変換を行っています．

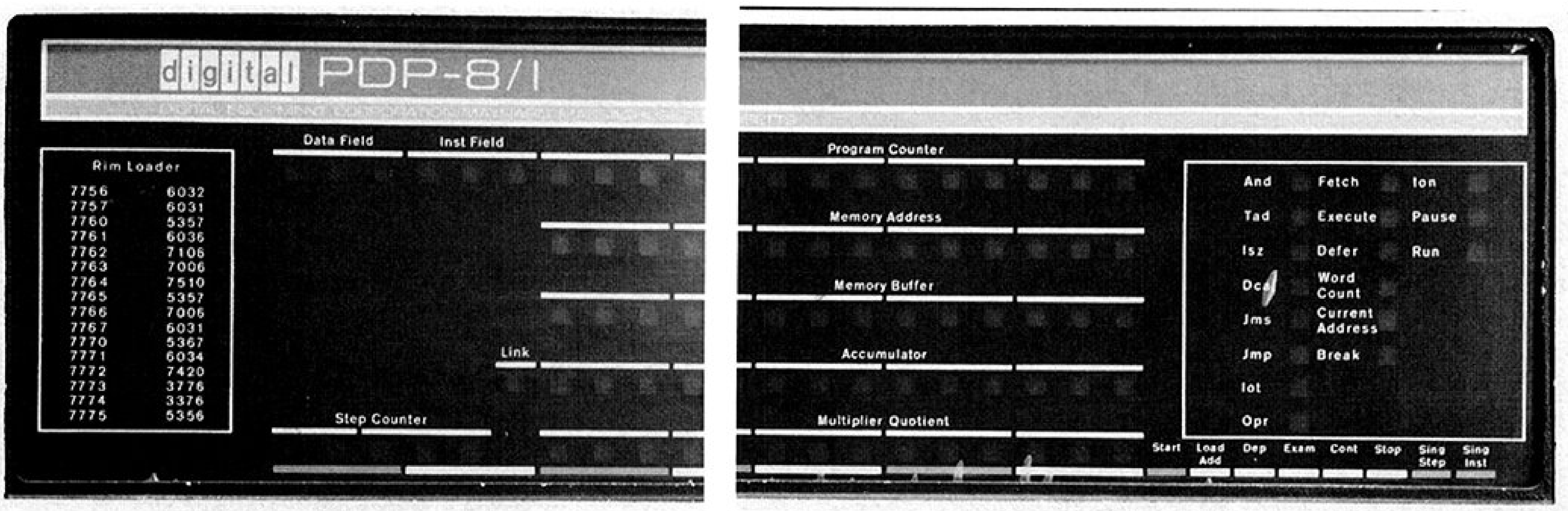
ほとんどメインテナンスフリーで，年に1回くらいカーボンの棒を点検し，摩耗していれば交換すればよいのです」

ニキデ・ン・ニーク 「勝男，テレタイプ以外の端末はないんですか？」

勝男 「LA30って言うのがもうすぐ出てきます．確か



〈写真4〉
PDP-8のオペレータコンソール



30字/秒の恐ろしく高速で，針のようなものを縦方向に何個か1列に並べ，左から右に動かしながら，それらの針をあるパターンで打つと文字が構成されるもので，ドットマトリクス方式と呼ばれ，ほどんど電子化されたすごいやつ，なんと重さはたったの50キロで僕と同じくらいしかないんですよ」

● PDP-8のビデオ端末

フュージット・フェイカム　「ビデオ端末はないのですか」

勝男　「まだ企業秘密なんですが，1年くらいで画期的性能を持ったものが発表されます．その名はVT05というもので，重さがたった25キロほどで簡単に持ち運べ，どこにでも設置でき，非常にコンパクトかつ高速表示が可能な，ちらつきがまったくないものが提供されます」

イターキ・イターク　「そんなに軽かったら盗まれて困るじゃないか，適当に重いからこそいいんじゃないか．まーいいか，じゃー別の質問だが，なぜビデオ端末がちらついたりちらつかなかったりするんだ？」

勝男　「PDP-12のビデオ表示を例に説明しましょう．PDP-12のビデオ端末は，入力はテレタイプから行い，CPUがその文字をディスプレイインストラクションを使用して文字の半分をドット方式で表示し，その後もう半分を表示して一つの文字を表しています．

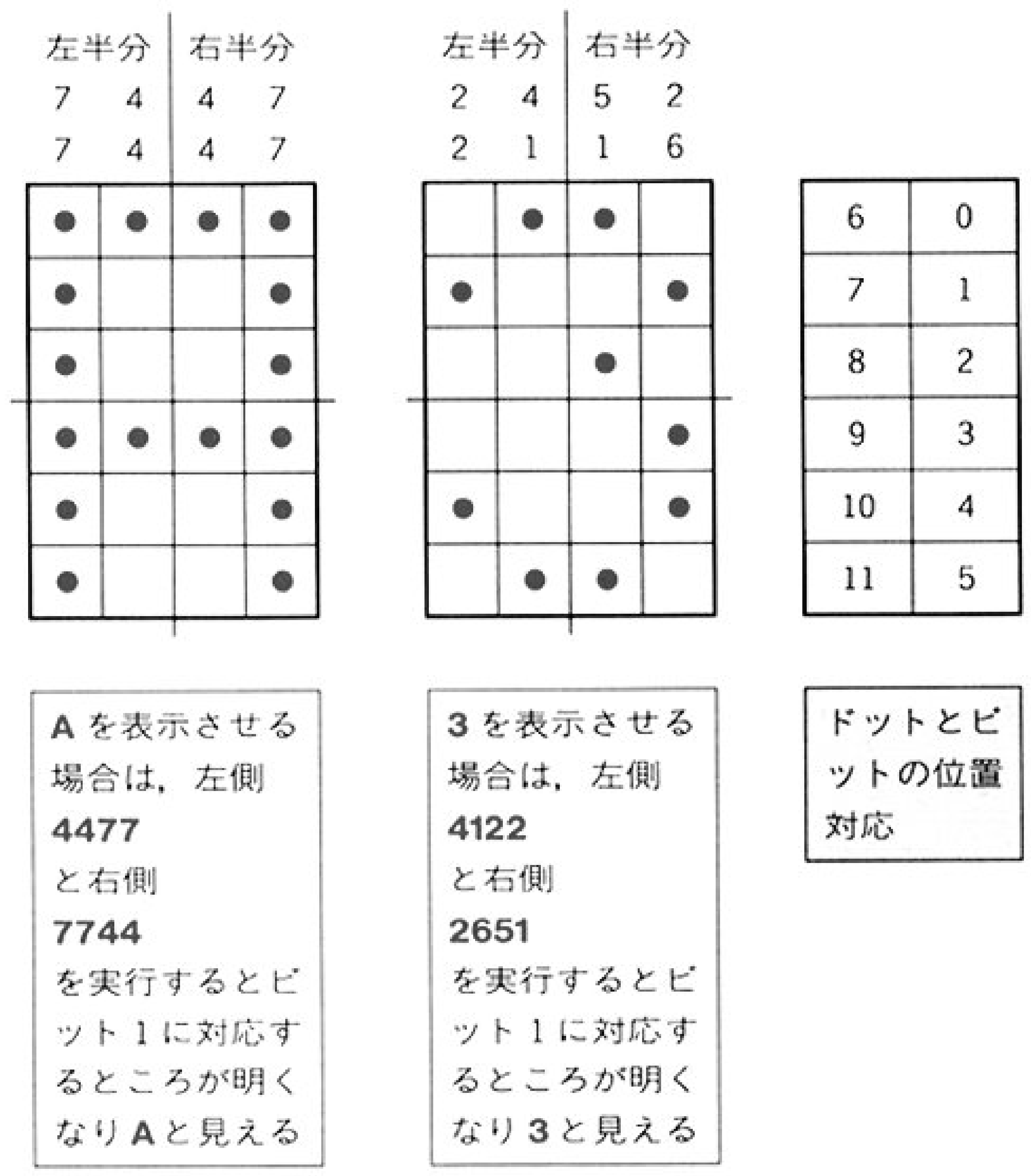
　このビデオ表示は残光時間が短いため，そのままにしておくと文字がすぐに消えてしまいます．そこで，何度も何度もCPUが表示を行うプログラムの実行を繰り返す必要があります．

　このため，あまり表示文字数が多くなるとCPUの負荷が大きくなり，PDP-12が速いといっても画面がちらついてくるのです．その表示方法の説明は**図2**のようになります．VT05は，これを数枚の内蔵ボードで行っているのでちらつかないのです」

● PDP-8の立ち上げ

　この説明の後，僕はシステムのブートについて話した．

〈図2〉PDP-12のビデオ表示



勝男　「システムのブートは非常に簡単で，まず手でオペレータコンソールからプライマリ・ブートストラップローダと呼ばれるわずか20ステップ弱の命令をメモリに入れ，テレタイプの紙テープ読み取り機にセカンダリ・ブートストラップローダ，磁気テープまたは磁気ディスク用のものをセットして，CPUをスタートさせます．

　すると，カチャカチャと音をたてながら紙テープを読み込みます．その後，所定のアドレスからCPUをスタートさせるとOSが立ち上がります．

　なんとも呆気にとられるほど簡単ですね．プライマリ・ブートストラップローダもわずか20ステップ弱なので簡単に覚えられます．

　何も見ずに素早く入れていくと，オー，こいつはす

〈図3〉
PDP-8の基本命令

0	1	2	3	4	5	6	7	8	9	10	11
OP code			I	C	直接アクセス可能メモリ						

OP code	0	**AND**(論理積)
	1	**TAD**(2の補数を用いた足し算)
	2	**ISZ**(Increment Skip on Zero. メモリを+1して0になったら次の命令をスキップ)
	3	**DCA**(Deposit Clear AC. ACの内容をメモリに書き込んだ後ACをゼロにする)
	4	**JMS**(Jump Subroutine. 指定したアドレスに，この**JMS**の次のアドレスを書き込み，指定したアドレスの次の命令から実行する)
	5	**JMP**(Jump. 指定されたアドレスにジャンプする)
	6	**I/O**(I/O命令．システムコンソールやディスク，磁気テープの制御を行う)
	7	**OPERATE**(演算の結果，ACがプラス，マイナス，ゼロなどのチェックを行い，その結果，次の命令をスキップする．ACを右/左に1または2個，ビットを移動させたりする．コンピュータ停止命令，ACの内容を+1する)
I		間接指定を意味する(そのアドレスの内容をアドレスとして利用する)
C		カレントページかページ0を指定する(**図5**). **0**：の場合は**0**番地から**177**番地(8進)までしか指定できない **1**：の場合はメモリを**128**番地(10進)ごとに区切った領域となる

〈図 4〉
JMS 命令の実行

アドレス	命令	
0100	JMS 150	(この命令の実行で制御が 150 番地に移る)
0101		
⋮		
0150	101	(JMS 150 が実行されると 150 番地に 0101 が格納される)
0151		
⋮		
01nn	JMP I 150	(この命令の実行で 0150 番地の内容をアドレスとして 0101 番地に制御が戻る)

ごいなんていわれますよ．是非覚えてください．それが面倒な人は，プライマリ・ブートストラップをダイオードマトリクス上でパターン化したオプションが提供されているので利用してください．一発でブートできますよ」

● 簡単に覚えられる命令

ひとまず説明が終わったので，次に最大のやま場である命令について説明することになった．

勝男 「PDP-8 は 12 ビットコンピュータで，その基本命令は**図 3** のようになっています．オペレーションコード(OPcode)と呼ばれるものが 3 ビットしかないので，命令の種類はなんと 8 種類ということになります．簡単に覚えられますので，是非暗記しておいてください」

そして，命令の説明を始めた．

勝男 「**AND** については言うまでもありませんね．ビットごとの論理積です．**TAD** は 2 の補数を用いた足し算で，桁上がりが発生しても後処理の必要はありません．**ISZ** はループの実行回数のチェックなどに用います．

JMS は**図 4** のように行われます．また，**JMP** は制御を強制的にアドレス部で指定された番地に移します．

I/O 命令は，ビット 3 からビット 8 までがデバイスの指定番号で，03 ならシステムコンソールのキーボード，04 はシステムコンソールのプリント，ビット 9 からビット 11 までが I/O 実行のためのパルスを発生します．**OPERATE** 命令は多くの論理演算を行うことができます」

一同納得したようだ．

勝男 「それでは，4 K ワード以上のメモリのアクセスはどのようにすると思いますか？ この仕掛けこそセグメント方式と呼ばれるもので，PDP-8 で採用しているものです．これは，後のマシンにも形を変えて応用されています．

まず，IF(Instruction Field)，DF(Data Field)と呼ばれるレジスタにより，最大 32 K ワードがアクセス可能になります．IF や DF を変えながら，4 K ワード単位で 32 K ワードのすべての領域にアクセスできます．

IF, DF は同じでも違っていても構いません．PDP-8 にはメモリ保護などはありませんから…単純にアドレスの拡大だけを可能にしました(**図 5**)．

では次に，この少ない命令をどのように用いていろいろなアプリケーションを作るのでしょうか．例えば，PDP-8 では掛け算，引き算，割り算，比較などの演算が用意されていません．ここで A 番地と B 番地の内容が等しいことを知るにはどのようにしたらよいのでしょうか？」

〈図 5〉 PDP-8 のメモリ管理

0〜4 K ワード時のページ番号

ページ	アドレス
カレントページ 37	7777(8) 7600(8)
カレントページ $n+1$	⋮
カレントページ n	⋮
カレントページ 1	377(8) 200(8)
ページ 0	177(8) 000(8)

4〜8 K ワード時のページ番号

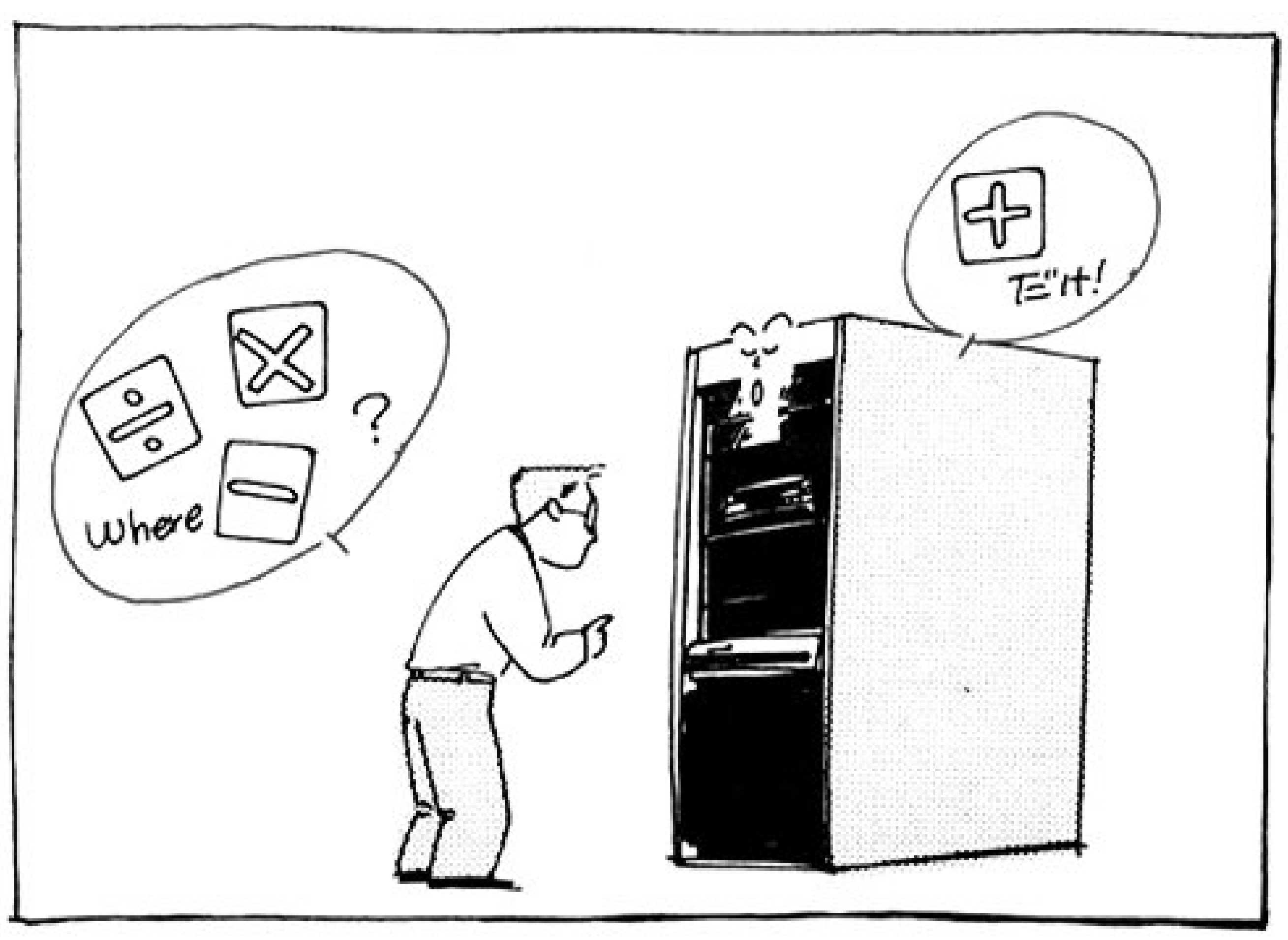
ページ	アドレス
カレントページ 77	17777(8) 17600(8)
カレントページ $4n+1$	⋮
カレントページ $4n$	⋮
カレントページ 41	10377(8) 10200(8)
ページ 40	10177(8) 10000(8)

メモリのページ割り付け

PDP-8 では，直接アクセス可能なアドレスは **000**(8) から **177**(8) までの 128 番地分しかありません．そこで，関接指定を使用して 4 K ワードまでのアドレスをアクセスできるようになっています(例：**TAD I 50** では，50 番地に **7756** などを入れておくことで 128 番地の境界を越えてデータをアクセスできる)．

4 K ワード以上のメモリアクセス

4 K ワード以上のメモリをアクセスするためにセグメント方式を使用しています．これには，**IF**(Instruction Field)，**DF**(Data Field)と呼ぶ各 3 ビットのレジスタを用います．**IF** と **DF** の値を変えながら 4 K ワード単位で最大 32 K ワードをアクセスできます．



〈図6〉
二つの値が等しいかどうかをPDP-8で求めるには

等しいということは，

① A番地をACにもってくる
② ACの1の補数をとる(すなわち0，1をひっくりかえす)
③ ACの内容を+1する(すなわち2の補数をとったことになる)
④ B番地の内容とAC間でTADを実行する
⑤ OPERATE命令でACがゼロかチェックする(ACがゼロならAとBは等しいということになる)

ではAとBに数字を入れて上記の方法を説明すると，

A=1234(8)，B=1234(8)として，			
① AC	=1234	TAD A	
② ACの補数	=6543	CMA	(オペレート命令で，Complement AC)
③ AC+1	=6544	IAC	(オペレート命令で，Increment AC)
④ AC+B番地の内容	=0000	TAD B	
⑤ ACがゼロ，つまりA=B		SZA	(オペレート命令で，Skip on Zero AC)

ということで，僕はニキーデ・ン・ニークに質問をしてみた．

ニキーデ・ン・ニーク　「そんなことはわからん，君達コンピュータ会社の人が考えてくれ．我々はそれを利用するだけだ」

僕はムッとして，

勝男　「なんでも一緒にやらなければ，一緒に大きくなっていけませんよ．DECだけでできることなんて限られてるし，IBMのような会社と違って，なんでも自分でできるんじゃないんだ．今のDECやOEMは相互の協力がなければ大きくなれないんですよ」

ニキーデ・ン・ニーク　「それもそうだな．我々ももっと勉強しなくちゃ．そして，もっといいものを作り出していきたいものですね」

と言ってくれたので，僕は今まで独立独歩の感じのした彼のことは好きではなかったが，彼もそんなに悪くないなと思った．

勝男　「では説明します．等しいということは，**図6**に説明する手順で行います」

一同真剣なまなざし．

勝男　「どうです．簡単にできたでしょう．では，引き算はどのようにするのでしょうか？ 15−3　はどのようにしたらよいと思いますか？(**図7**)」

一同真剣なまなざし．

勝男　「この方法は10進法でもできますので，やってみましょう」

15−3=15+(各桁9と3の差96に1を加える．10の補数に相当する)
=15+97
=12(最上位の桁上がりは無視する)

勝男　「では3−15はどうするんでしょう．桁上がりに注意して試してみてください．そのほかにも2をかける場合はACを左にシフトし，2で割る場合はACを右にシフトします．では13×5はどのようにしたらよいか？ ちょっと考えてみてください．コンピュータの演算のしくみがわかってきますよ．

こんな簡単なことでも知恵を絞りながらプログラミングできるなんて楽しいじゃないですか．それも，使えるメモリが限られ，使える演算も限られた中で非常に多くのことをコンピュータで行えるなんて素晴らしいことじゃないですか．そう思いません？

では，ここいらで昼休みにしましょう．午後は1時から始めますので遅れないように．君達アメリカ人は時間にルーズなところがいけませんね」

と注意を喚起してランチにした．

〈図7〉PDP-8で15−3を行うには

−3は，**0011**の2の補数をとり**1101**となる
(1，0を反転させ1を加える)

```
      1111(15)
  +   1101(−3)
  ------------
 1←  1100(12)
```

2の補数なので桁上がりは無視する．
よって答えは12．
1の補数の場合は桁上がりは最下位桁に加える

割り込みってなに？

案の定，全員が遅れた．彼らときたら，まっ昼間から水がわりにビールを飲むから困ったもんだ，と思いながら午後の授業を始めた．

勝男　「オバタリアンという日本で有名な怪物がいますが，割り込みとはスーパーマーケットなんかで，並んでる人を無視して行列に強引に入り，注意すると『あっら一並んでらっしゃるのぉ～』なんて言うのがいますが，まさにあれです．

そのほかに，軽四輪を運転してて右にいっぱいハンドルをきってセンターラインをはるかに越えて左に曲がったり…これも割り込みです．まったく『君の車は大型バスかと言いたくもなる，僕の新車どうしてくれる』…」

〈図 8〉
PDP-8 の割り込み
(①～⑥の順で処理が行われる)

アドレス	処理内容
0000	②：**JMS0** 実行の結果，ここに **1001** が格納される
0001	③：I/O 命令でどのデバイスが割り込みを発生したかチェックし，処理ルーチン **NNNN** へジャンプする
⋮	
NNNN	④：処理ルーチンの先頭アドレス
⋮	⑤：**JMP I 0** を実行することで **1001** 番地にジャンプする
1000 命令 1	①：割り込み発生→命令 1 を実行後ハードウェアが **JMS0** を実行
1001 命令 2	⑥：すなわち割り込みの発生した次のアドレスに戻ってきた



フェイカム 「勝男，どうしたんだい，軽四と割り込みと何の関係があるんだい，オバタリアンに恨みでもあるのかい？」

勝男 「すまない，私情をはさんで．割り込みになると昔を思い出して．で，コンピュータの割り込みとは実行中の作業を強制的に中断させ，別の作業をさせることですね．コンピュータの割り込みは別の作業をやった後は，ちゃんとすべての情報を中断した状態にまで戻すことまで責任を持ってくれるところがオバタリアンとは違いますね．

現在のシステムは，非常に多くのデバイスをユーザープログラムの実行順序を意識させることなくサポートしていますが，PDP-8 では割り込み処理をどのようにしてるか説明します．

なんと PDP-8 では，すべての割り込みは **JMS0** 命令をハードウェアが発行して処理しています．割り込みの優先順位は，ユーザーがプログラムの中で優先順位の高い順に I/O 命令を用いてデバイスのフラグ，これは終了/エラーを示すフリップフロップですが，これをチェックしています．では，**図 8** を使って割り込みについて説明しましょう」

勝男 「どうです．非常に簡単な割り込み処理機能でしょう．何か質問は？」

するとアムダンが聞いてきた．

アムダン 「勝男，割り込み処理で，何かのノイズでフラグが立ったらどうなるのかね？」

勝男 「PDP-8 の場合，フラグが Done か Error を代表してるのでそれらをチェックして何もなかったら，何事もなく元に戻ってきます．すなわち，何もやらなかったことと同じです．

見方を変えると非常にルーズなやり方に見えますがシステムが止まるよりはいいんじゃないですか」

長い 1 日が終わり帰れると思った時，

アムダン 「勝男，DEC ではミニコンがどうのこうのと言うからには，大型コンピュータがどんなものかは知ってて言ってるんでしょうね？ チョットさわりだけでも教えてくれないか？ IBM がどうのこうのという話をしてくれないか？」

なっ，なんだこの男は．こんな痛いところをついてくるなんて，もう時間がないし，断ろうかな？と思っていると，

アムダン 「頼むよ．勝男は DEC のことは教えてくれたけど，IBM の話があってもいいじゃないか？」

もう断れない，もうやるしかない．

勝男 「じゃー，あまり時間もないのでほんとに簡単にね」

と言うことでまた話が始まった．

勝男 「IBM が他社の追随を許さぬ圧倒的な地位を築いたのは，1960 年に発表した廉価版 1400 シリーズ（加減算約 0.3 ミリ秒，乗除算約 2.4 ミリ秒）で，爆発的な売り上げを記録してからなんだ．1964 年に発表した有名な 360 シリーズで，ついにファミリシステムの幕開けを築いたことは皆さんもどこかで名前だけでも聞いたことがあるでしょう」



勝男　「では，360の基本的な説明をします．まず名前の由来ですが，ものの本によると360度全方位カバーできるところからきてるらしいのですが，僕の考えでは，1年365日のうち最低でも360日は動作を保証できる高信頼性を誇ると言うことじゃないかな．『じゃ～，発表まじかの噂のある370はどうなんだ？』とくるかもしれんが，僕の説では，あと5日分の保守契約費用を負担してくれたら1年中の動作保証があると言うことだと思うんですが」

(編注：すごい皮肉)

勝男　「で，話を元に戻してIBM360は4バイトが基本の32ビットマシンです．24ビットアドレス，高速のチャネル，ミニコン以上に複雑な装置を介して高速な磁気ドラム，磁気ディスク，磁気テープ，ラインプリンタなどを接続しています．

チャネルはメモリとCPUを同じバスに接続して，メモリ上にチャネルが理解できるプログラムを格納しておき，CPUがSTART I/Oを発行します．

START I/Oを発行した後，CPUはほかの仕事を続け，チャネルはコマンドアドレス・ワードをもとにチャネルプログラムがあるメモリ領域を読み込んでチャネルプログラムを解読し，入出力制御装置に動作を促すという手順を踏んで入出力装置とメモリとの間でデータ転送が行われます．

バススピードは約5Mバイト/秒と高速で，かつ並列動作が可能です．

なんと，マイクロコードをROMの中に格納し，ハードウェアのコンパクト化を図っています．レジスタは32ビットが16個でR0はアドレシングモード時はちょっと特別な使用法があります．

浮動小数点用には64ビットのレジスタが4個あります．そして，五つの可変長な命令形式をもっています(**図9**)．

それから大事なことですが，オペレーティングシステムはOS360と呼ばれ，基本的には三つのバリエーションがあります」

▶ 32Kバイト以上あれば動くと言われているもので，一つのジョブのみが実記憶上で動作する．

▶ 64Kバイト以上あれば動くと言われているもので，実記憶上であらかじめ決められた固定数のジョブのみが動作する．

▶ 256Kバイト以上あれば動くと言われているもので，実記憶上で可変数のジョブがディスクに対してスワップイン，スワップアウトを行いながら動作する．

勝男　「非常にすっきりした型式だと思うが僕はあんまり好きじゃないな．どうしてかって，それはソースとディスティネーションの構成が対称でないのが気にいらないんだよ．

でも，データの移動なんかは**Displace**と**Len**を用いてブロック単位でできるなんて素晴らしいアイデアですね．

それに，マイクロプログラミングの考え方がIBM360をシリーズ化するのにたいへん有効な考えではないかと思うけどどうでしょう」

イブム・アムダン　「その意見には賛成だね」

彼とはその後もIBMのマシンやいろいろなコンピュータの構造などについて意見を交換した．

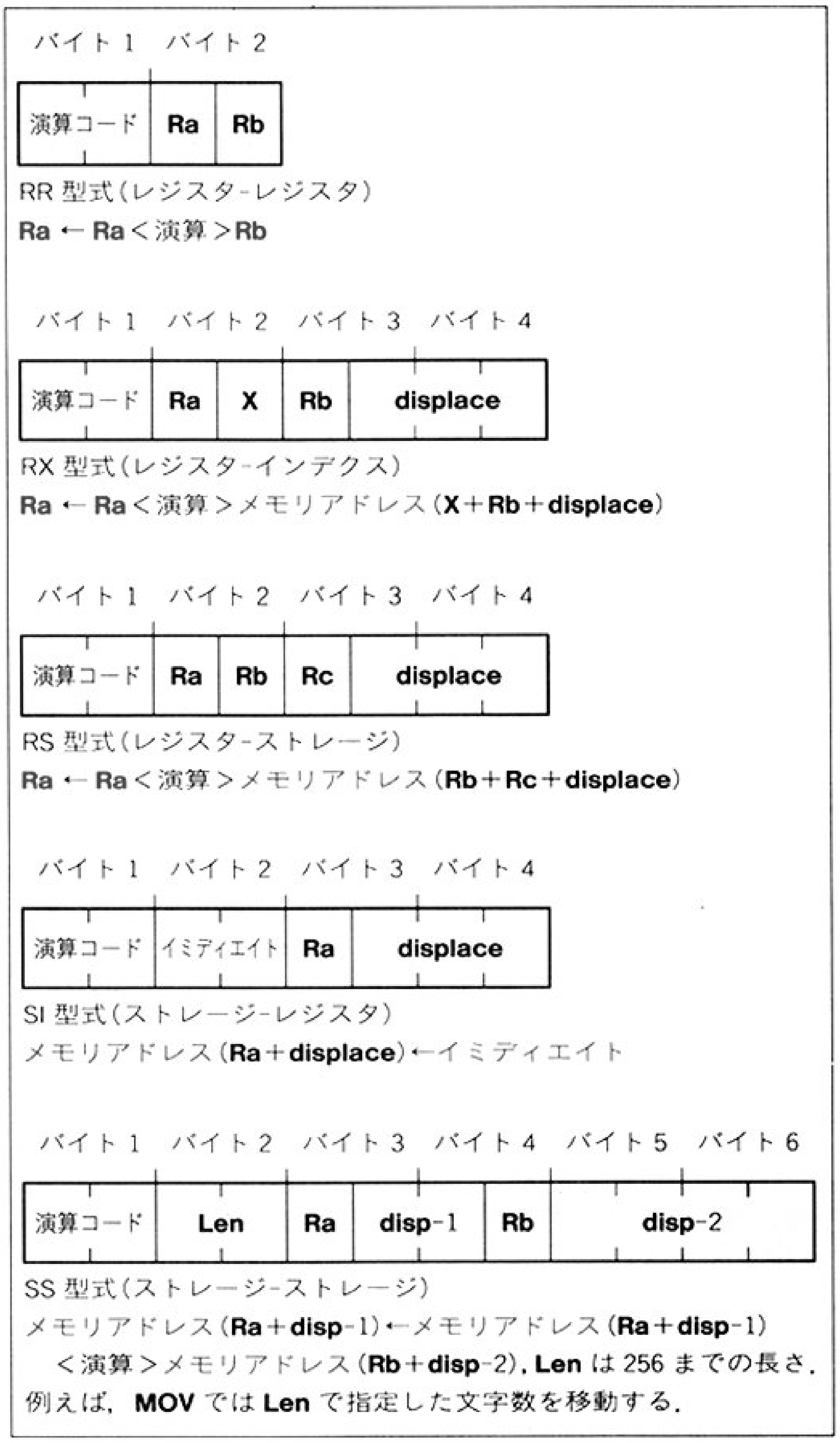
勝男　「何か質問はありますか？ たいへん短いお話でしたが，これにて終わりとします」

アムダン　「私はIBM360マシンは産みの親みたいによく知ってるんで聞いてみたんだが，まぁ～，いつかもっと突っ込んだ議論をしたいね」

と言うことで，みんなに別れを告げたが，どこかで会ったような，聞いたような，何かひっかかる思いがこの4人に対して残ったのは僕一人だけであろうか？

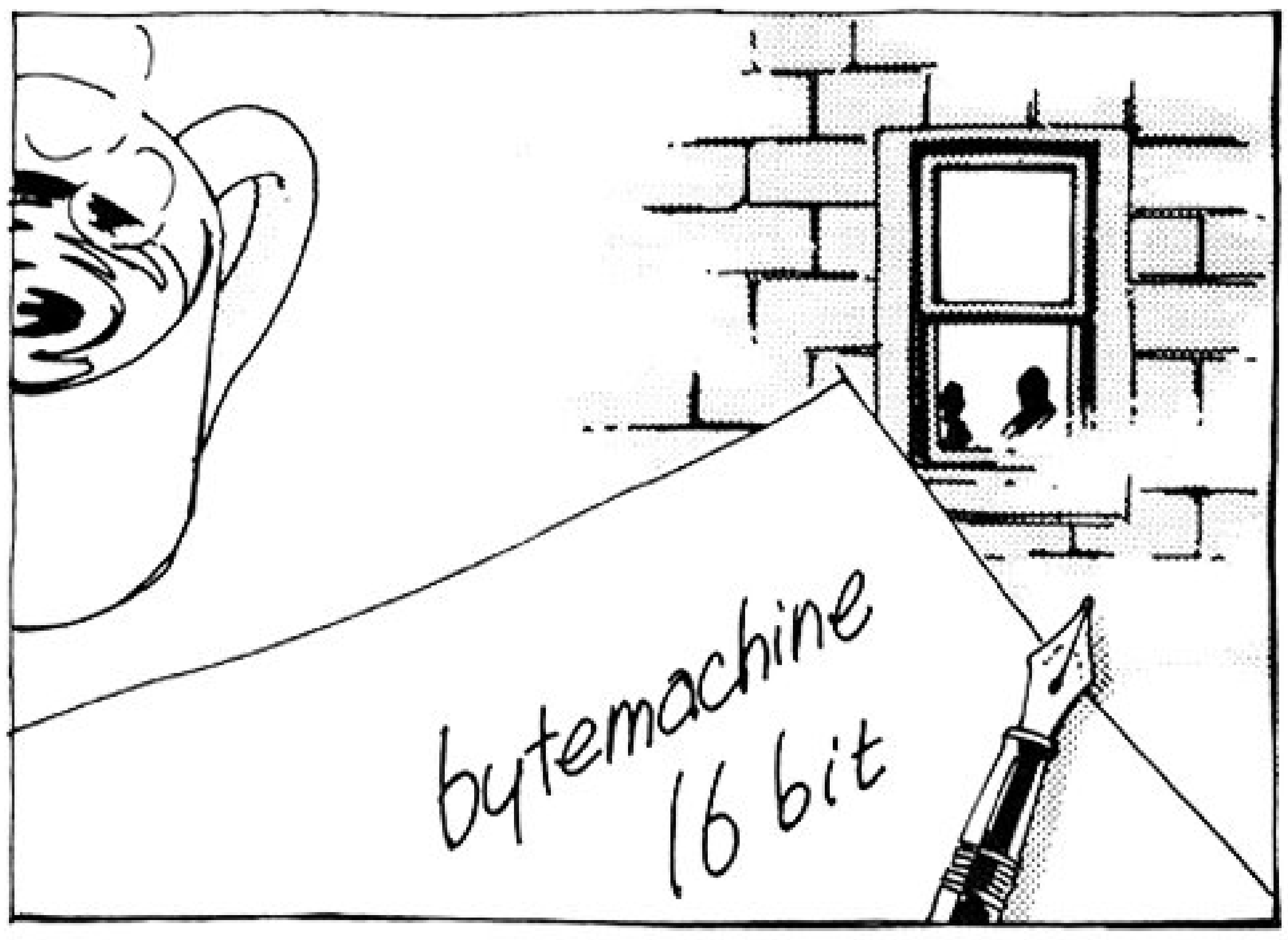
後で彼らの名前をもう少し注意深く眺めれば思い出すかもしれないと思った．

〈図9〉IBM 360の命令



PDP-11 シリーズの誕生とそのアーキテクチャ

ある霧深い朝，ケンと新しいコンピュータのアーキテクチャについて議論を再開した．



ケン　「今度のマシンは 16 ビットのバイトマシンにしたいと思うんだ．IBM もシステム 370 のバイトマシンを発表しているし，他のメーカーも右へならえで 8 ビットの整数倍を単位としたコンピュータを出している．

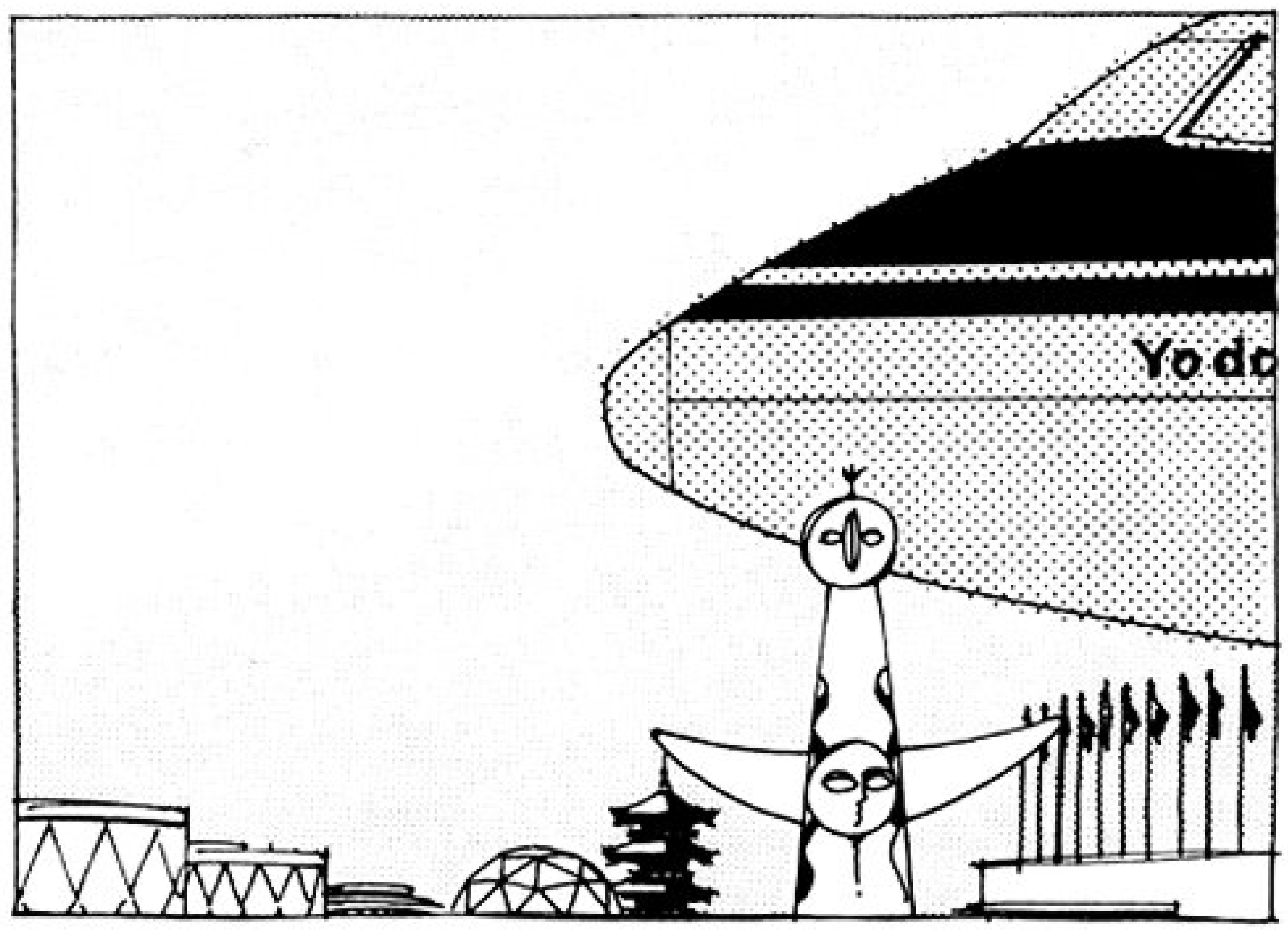
DEC は今までワードマシンを出してうまくやってこれたけど，これからの時代の流れを考えると，アーキテクチャを一新したミニコンを開発する必要があるのはまちがいない．

そこで，レジスタの長さを 16 ビットで 8 個用意し，複雑なアドレス演算やいろいろな算術演算を行えるモードがあって，おまけにユーザーデバイスが簡単に使えて拡張性のあるものがいいだろう」



勝男　「それは，すごい．うん．それではケン，こんなのはどう．CPU，メモリ，各種デバイスなどをほとんど同格に扱い，おのおの独立で動作できるようにするんだ．それが 1 本のケーブルで簡単に接続できるなんてのはどうかな？ それも将来性を見越したバス構造にするんだ．そうだね，スピードは 2 M バイト/秒もあれば大丈夫だと思うよ．それにメモリは 256 K バイトくらいつなげたら当分の技術革新にはついていけると思うんですが？」

と言うことで，さっそくデザインが始められスピードは 1.5 M バイト/秒，メモリアドレスは 256 K バイト，データ幅は 16 ビットに決まった．ときに 1970 年のことだった．



あろうことか，以前 PDP-8 の説明をした会社が，今度は新 16 ビットマシンをこれからの目玉にするということなので，新しいコンピュータの説明をすることになった．

例の 4 人はまだいるかなと思いつつ会議室に入ると，いたいた，顔見知りがたくさん．

勝男　「どうだい調子は？」

と聞くと，みんな訛りのあることばで，

エド　「いやー，ほんまに PDP-8 は売れて売れてたまらんわ．うちは歩合制やからこたえられんわ．会社でもはじめようかと思うんだが，勝男はどうだい？」

勝男　「万事順調だけど，給料が上がらないからつらいよ」

と答えて説明を始めることにした．

勝男　「こんど DEC から NOVA に続く世界 2 番目の 16 ビットミニコンピュータを発表します．このシステムのために I/O デバイスの拡張に関して非常に強力なバスを開発し UNIBUS(ユニバス)と命名しました．

ユニバスは最大 15 メートル，56 本の信号線と各種電源線，グラウンド合わせて 144 本のピンをもつ柔軟性のあるケーブル構造にしました．また，ユニバスの仕様はオープンにし，いろいろなメーカーが周辺機器の分野に参入しやすくする方法を取っています．

この 16 ビットマシンは，まだ使っていない数字 11 を使い PDP-11 と命名しました．では具体的に PDP-11 のブロック図を**図10** に示します」

なんとプログラムカウンタまでもが汎用レジスタだった

● PDP-11 の内部構造

今度は PDP-11 の内部構造について説明した．

勝男　「皆さん，このシステムでは**図10** のように，I/O デバイス制御用レジスタをメモリ空間の一部としてアドレスに割り付けています．この方法だと，実際のメモリが I/O のアドレスに専有されてしまいますが，それでも I/O の扱いやすさがそれを上まわります．上位 4 K ワードがそれに当てられます．

また，プログラムカウンタも汎用レジスタ，および I/O アドレスの一部になっています．この結果，プログラムカウンタも演算の対象になり，通常の倍のアドレス演算が可能になっています．

汎用レジスタは R0～R7 まであり，R6 がスタックポインタ(SP)，R7 がプログラムカウンタ(PC)です．汎用レジスタには，177700～177707(８進)までのアドレスが割り当てられています．割り込みも，デバイス特有のアドレスを持ちベクターアドレスと呼びます．

PDP-11 には PDP-8 にあったような I/O 命令がありません．では，どのようにしてデバイスの制御を行うのかですが，メモリマップト I/O と呼ばれる方式を採用しています．ちなみに，PDP-8 は I/O マップト I/O といいます．

つまり，I/O デバイス制御用レジスタには，メモリと同じようにアドレスが付けられているので，**MOV** 命令でこのレジスタにデータを書き込んだり，読み出した内容を **TEST** 命令でチェックをして制御します．

例えば，システムコンソールに“A”を表示したいときには，オペレータコンソールから 177566 番地に 101(8 進，16 進では 41)を書き込むか，

MOVB #101, @#177566

(**MOVB** 命令でコンソールの出力用レジスタのアドレス 177566 番地に 101 を送る)

という命令を実行すればよいのです」

一同納得．

〈図11〉 PDP-11 の命令形式

PDP-11 のダブルオペランド命令形式

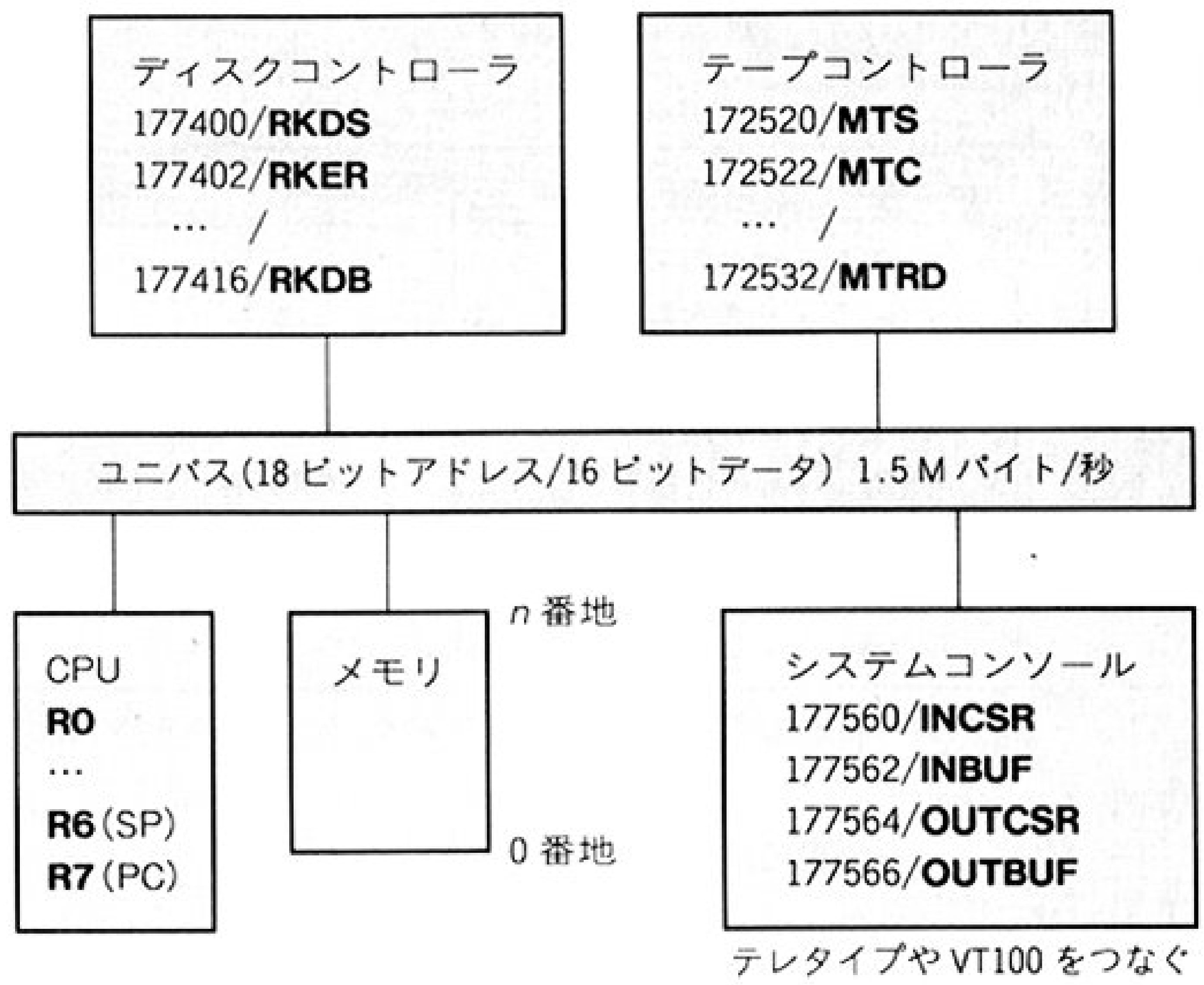
5	4	3	2	1	0	9	8	7	6	5	4	3	2	1	0
B	OPR			Smode			SR			Dmode			DR		

PDP-11 のシングルオペランド命令形式

5	4	3	2	1	0	9	8	7	6	5	4	3	2	1	0
B	OPR									Dmode			DR		

B：０の時はワード命令，１の時はバイト命令
OPR：オペレーションコード　Smode：ソースモード
SR：ソースレジスタ　Dmode：デスティネーションモード
DR：デスティネーションレジスタ

〈図10〉 PDP-11 のブロック図

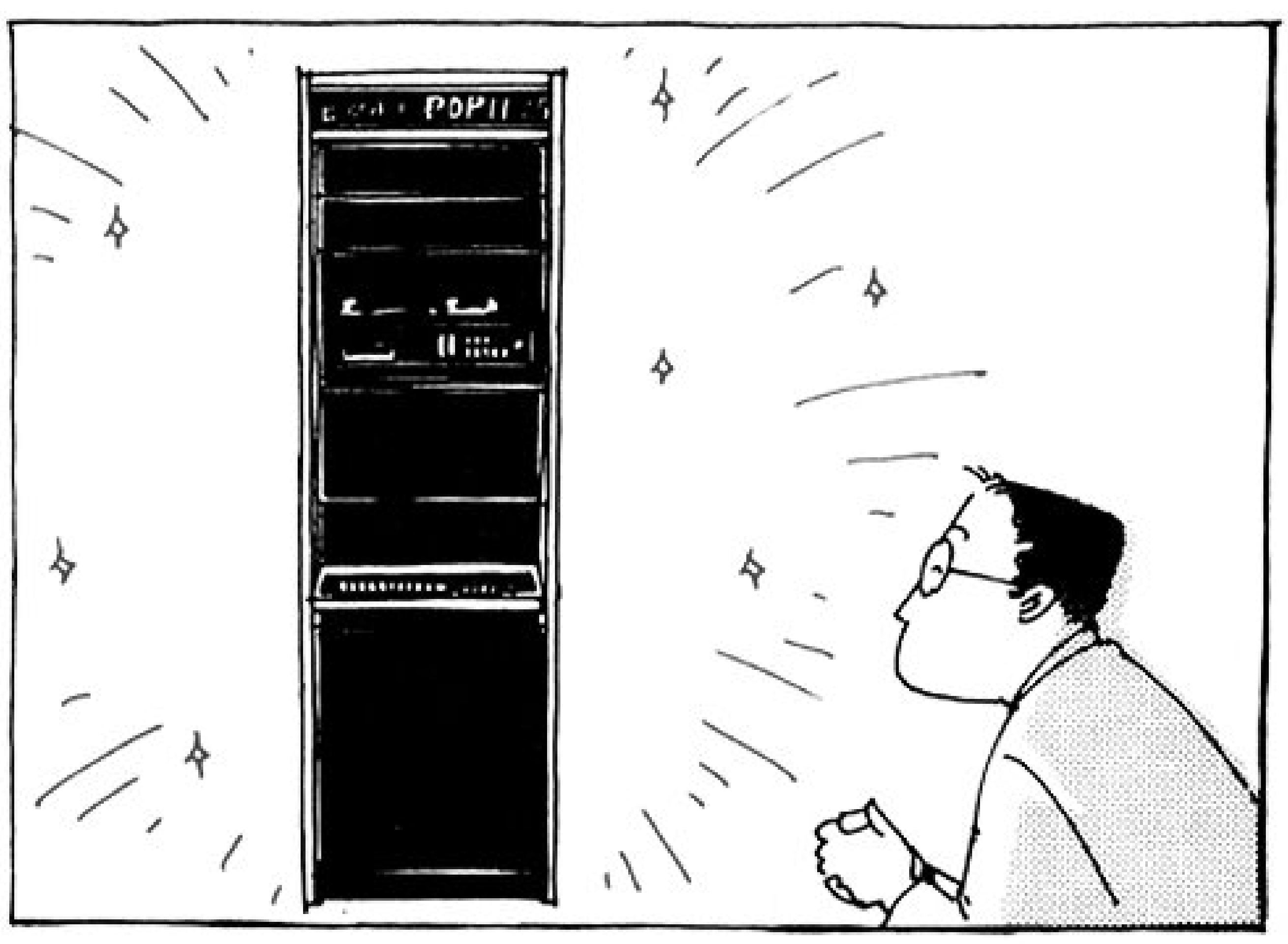


INCSR	：インプット用コントロール・ステータスレジスタ（キー入力するとフラグが１になる）
INBUF	：インプットバッファ(キー入力された文字が入る)
OUTCSR	：アウトプット用コントロール・ステータスレジスタ（キー出力するとフラグが１になる）
OUTBUF	：アウトプットバッファ(印字したい文字を入れる)
RKDS	：ドライブステータス
RKER	：エラーレジスタ
RKDB	：データバッファ
MTS	：テープステータス
MTC	：テープコントロール
MTRD	：リードデータ

● PDP-11 の命令形式

勝男　「PDP-11 の命令は 8 進法で区切られており，かつオペコードや各種モードが 3 ビットごとに分けられているので，わかりやすくて覚えやすい構造ですね．

では次に PDP-11 の命令形式について説明します(**図11**)．これら以外にもオペレーション，ブランチなどの種類があります」



勝男　「ソースはデータの発生元，デスティネーションは送り先を意味します．では，インストラクションについてもう少し詳しく見てみましょう．まず，レジス

〈図12〉 R0～R6 のアドレッシング

Mode	R0～R6

モード	シンボル	動作内容
0	R	R の内容がオペランド
1	(R)	R の内容がアドレス
2	(R)＋	R の内容がアドレス．その後 R＋(1 または 2)[注]
3	@(R)＋	R の内容がアドレスのアドレス．その後 R＋2
4	－(R)	R の内容を(－1 または－2)．R の内容がアドレス[注]
5	@－(R)	R の内容を－2．R の内容がアドレスのアドレス
6	X(R)	R の内容＋X がアドレス
7	@X(R)	R の内容＋X がアドレスのアドレス

（注） バイトの時は＋1．ワードの時は＋2 する．マイナスの時も同様

〈図13〉 R7 のアドレッシング

Mode	R7

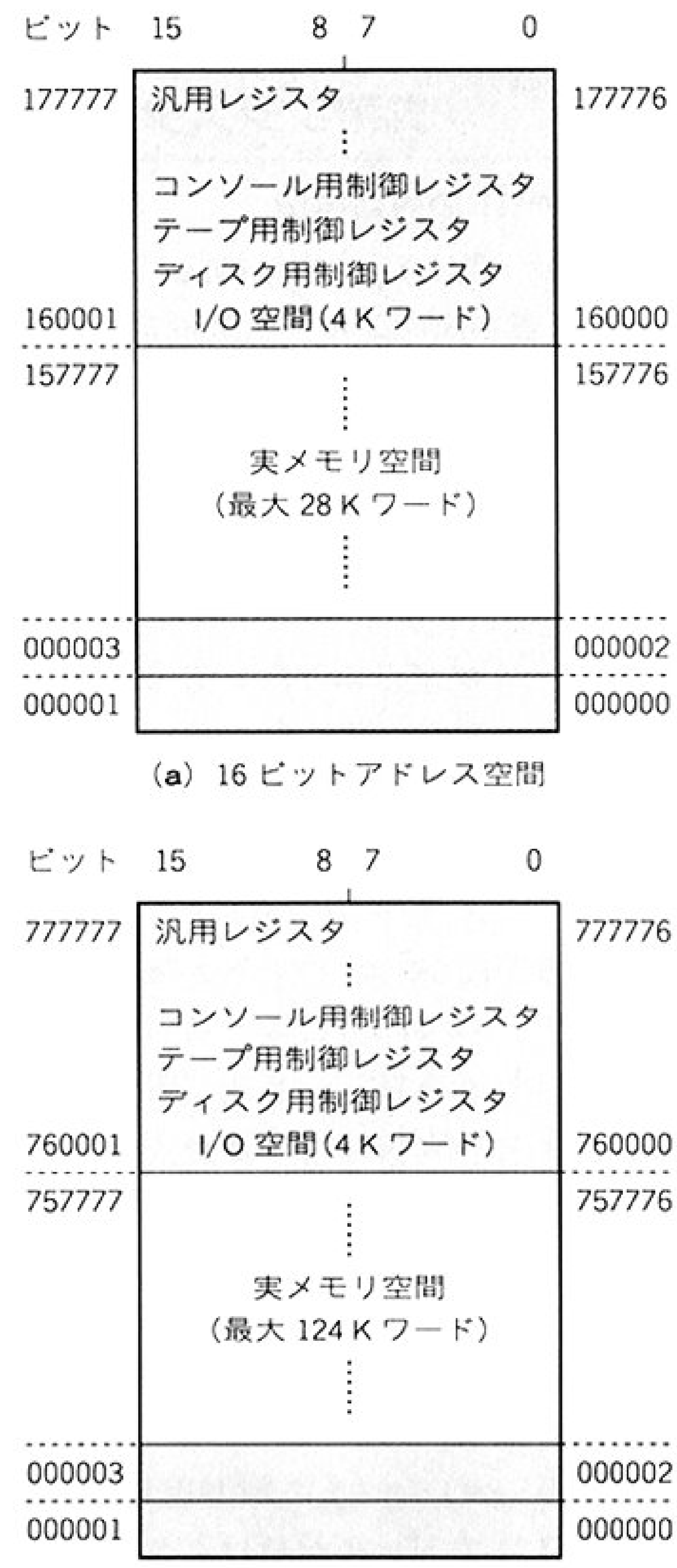
モード	シンボル	動作内容
2	#n	オペランド n は，この命令の次にある
3	@#A	アドレス A はこの命令の次にある
6	A	命令アドレス＋4＋X がアドレス
7	@A	命令アドレス＋4＋X がアドレスのアドレス

▶ この場合をプログラムカウンタ・アドレシングと言う

〈図14〉 MOV #101, @#177566

アドレス	内容
1000	112737 ; MOVB #101, @#177566
1002	000101 ;
1004	177566 ;

〈図15〉
PDP-11 の
アドレス空間



(a) 16 ビットアドレス空間

(b) 18 ビットアドレス空間

タが R7(PC)以外の場合を**図12** にまとめました」

勝男 「次に R が PC の時についてです．この場合はモードが四つしか存在しません(**図13**)．その理由は皆さんで考えてみてください．また，なぜ動作が**図13** のようになるかですが，PC は命令を読み出した時はすでに＋2 されているからです，そのことを上記のモードの動作内容に対して適用すると理由がわかります」

一同納得．

勝男 「ではここで 1000 番地に，

MOVB #101, @#177566

があるとして，これを機械語に翻訳してみましょう(**図14**)．どうです，簡単でしょう．16 進表記ならこんなに簡単にはいきませんね」

● PDP-11 のアドレス空間

勝男 「PDP-11 では，16 ビットアドレスのすべてを指定に用いると 32 K ワードまでアクセスできます．アドレス空間は，**図15** のように割り当てられています．PDP-8 とは異なり，ビットの位置は右から左に向かい 0 から 15 となります．

アドレスはバイト単位で割り振られています．32 K ワードというメモリは十分に大きなものだと思います．また，浮動小数点もオプションでサポートしたので，商業，工業，科学研究などの分野にも使用できると思います．PDP-8 と互換性はありませんが，飛躍的な性能アップでしょう．**図15** をワードでアドレスを見ると最大値は 177776 となりバイトで見ると 177777 となります」

一同納得．

勝男 「ここで一つ困ったことが起こります．PDP-11 は 16 ビットマシンなので，メモリにアクセスする時，

MOV #101, @#777566

なんて命令をメモリに書くことはできません．なぜならこの数字は 18 ビットですから．

MOV #101, @#177566

とすると問題が発生しません．アドレス 16xxxx 以上は 76xxxx と変換されてユニバスに出力されるからです．

僕の予想では 124 K ワードのメモリを使うアプリケーションはそうないのではと思っています」

とすかさず，

フェイカム 「それ以上のメモリが必要になったらど

うするんだ？」

これに対して，

勝男　「メモリマネジメントの拡張部分(PDP-8のIF/DFのような部分)を大きくして対応するでしょうね．数Mバイトくらいまではできると思います．アドレスのマッピングがたいへんになりますが．

でも，たったこれだけのやり方でアドレスの拡大ができるなんて画期的ですね．では，皆さんのビジネスの成功を確信してます」

そして，僕は彼らに別れを告げた．

● **PDP-11の成功**

1970年PDP-11の最初の製品PDP-11/20を発表した後，ケンに向かって言った．

勝男　「僕は確信してるよ．絶対にうまくいく．このシステムは，後世に残る名機になるよ．ケンも最高の経営者になるだろう．すべてうまくいくさ」

その後のPDP-11の売れ行きは爆発的で，DECも急成長した．しかし，それからの数年間のメモリの技術は我々の予想をはるかに越えるもので，あっという間にユニバスのアドレスを使い切ってしまった．

アドレスの限界とシステムの高速化を解決する手立てを考え，ついにメモリをユニバスから切り離し，CPUに直結のメモリバスで制御し，アドレスのセグメント化でメモリの拡充を図るということを繰り返し，1975年にPDP-11/70を発表した時には「16ビットでは限界だ」ということをみんなが感じ取っていた．

PDP-11のアドレス拡大，もっとメモリを（複雑なセグメント化への道）

ではなぜ16ビットでは限界なのか…PDP-11/70を例に説明しましょう．

PDPシリーズがいろいろな分野で利用されるようになると，より多くのメモリが必要となりました．

最初は124Kワード(18ビットアドレス)まで推し進め，これと上位互換性を保ちつつ，ついに2Mワード(22ビットアドレス)まで拡大してきました．

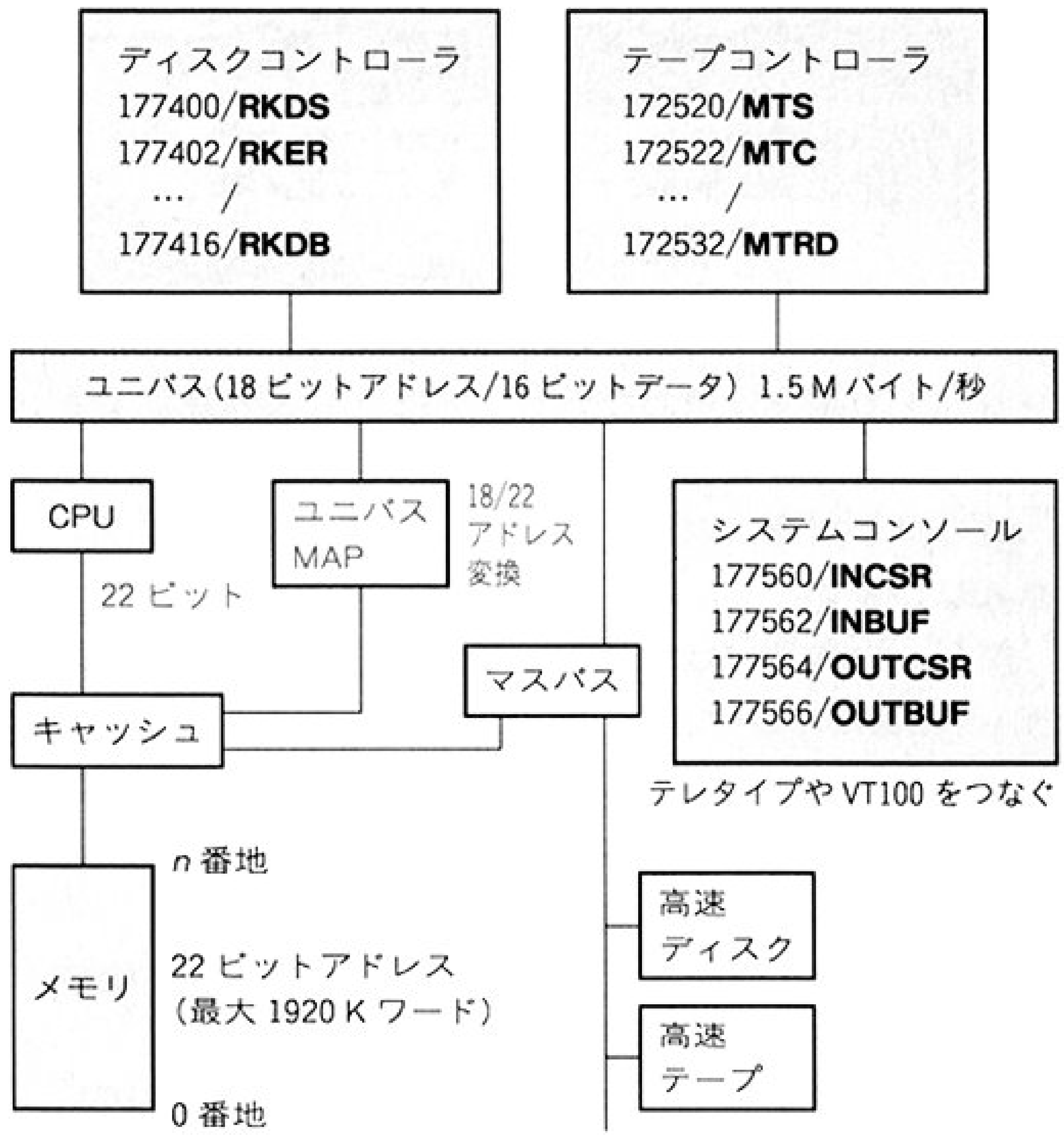
この方法は，基本的に16ビットのアドレスに最大16ビットのページレジスタを重複させ，加算して22ビットアドレスを得ます．この方法はIntelの8086に採用されたセグメント方式によく似ています．

PDP-11ではマルチユーザー，タイムシェアリングを可能にするためにメモリ保護などの機構があり，カーネル，スーパーバイザー，ユーザーの三つのモードがサポートされています．

PDP-11は高速化，大容量化するにつれて従来はシステムバスであったユニバスが，メモリを切り放し，I/O専用のバスになりました．

そして，メモリにはもっと高速の専用バスを設けました．124Kワード以上はユニバスには接続できません．

〈図16〉PDP-11/70のブロック図



PDP-11/70においては，ユニバスはI/O専用となりメモリはCPUに直結されています．アドレス構造がいかにもいびつになっているのが，**図16**からもわかると思います．

なぜいびつか？

それは，ユニバスからDMAを用いても，18ビットでは22ビットのメモリ領域を指定できないので，ユニバスMAPというアドレス変換機能を用いて22ビットに変換する必要があるのです．

また，PCは16ビットなのにメモリは22ビットで，ここでも変換が必要になります．16/18/22ビットのどのモードで動いてるかということも，アドレス変換に影響します．さらに考えると，ほかにもいろいろな問題がでてきます．

そしてPDP-11は，この後メモリアドレスが22ビットのままで，小型化を追及する方向に向かいました．



スーパーミニコン VAX11/780 の出現

32 ビットスーパーミニコンは驚異のマシン

ケン 「勝男，もう PDP ではこれ以上大きなアプリケーションの開発は望めないし，メモリも大容量にするにはアドレスの限界がある．32 ビットに移行しよう．一時も早く開発に取りかかろう．もう時代は 32 ビットを欲してるよ」

僕もこの意見には賛成だったので，早速アーキテクチャの検討に入った．

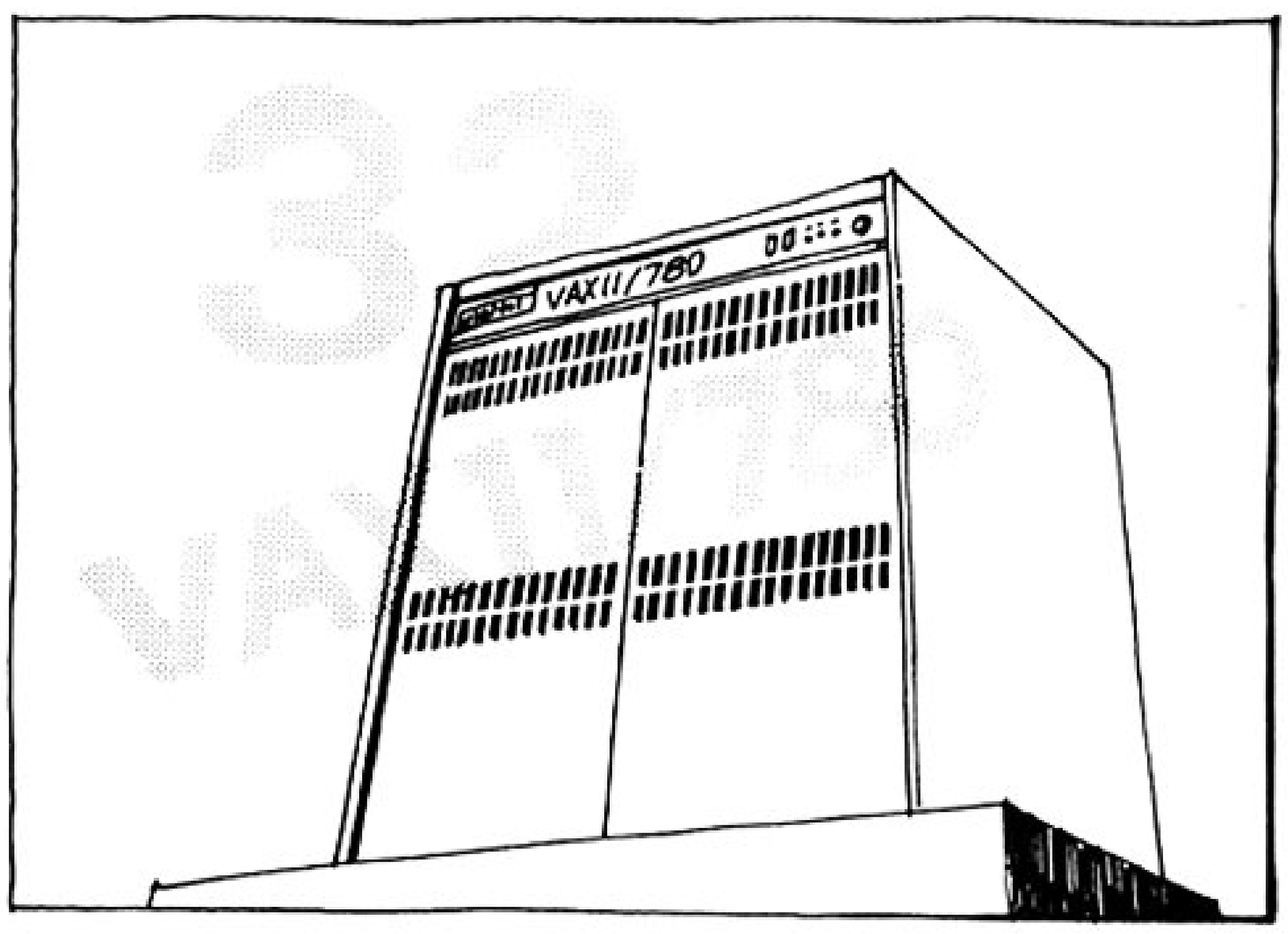
まず，どのような記憶方式を採用したらよいか，これには 32 ビットアドレスを四つに分けて，これらを実メモリに対応させる仮想記憶方式を採用した．

また，どのようなレジスタ型式，どの程度の性能のバスにするかなどの検討に入った．

このプロジェクト名は“STAR”と決定し，CPU，メモリ，ユニバスアダプタ，マスバスアダプタ，クラスタコントロールアダプタが一つの高速なバスを介して，ゆるやかに結合する方式のものにした．

そして OS の開発は，PDP 上のエミュレータを用いて行い，1977 年，ついに 32 ビットスーパーミニコン VAX11/780 が発表された(**写真 5**，**図17**)．

VAX は Virtual Address eXtention の略で，OS は VMS(Virtual Memory System)と命名された．



このマシンは，PDP-11 のように小さいアドレスをメモリマネジメント機構でゲタをはかせて，いびつに拡大するのではなく，32 ビットという広大な仮想の空間でプログラムを考え，これを実在する実メモリに 512 バイトのページ単位に対応させながら動作させる仮想記憶方式を採用しています．

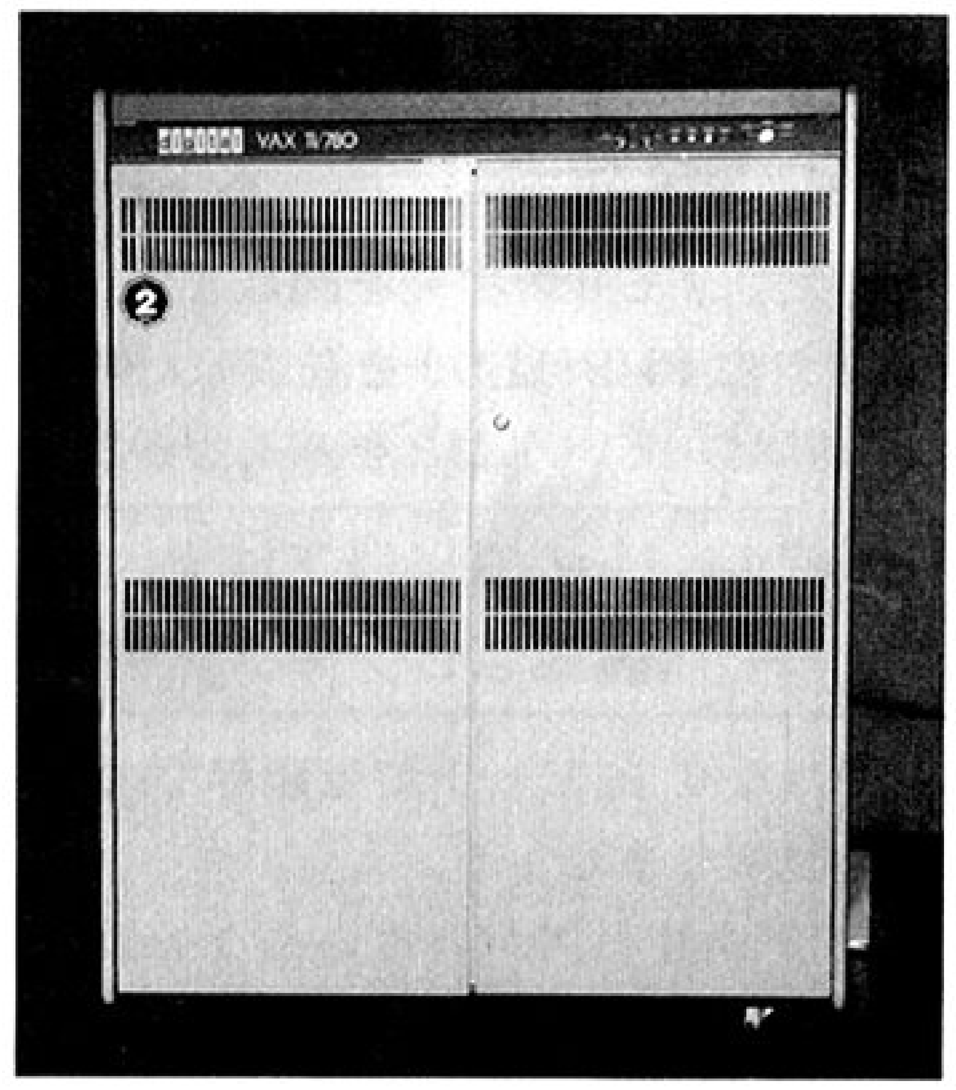
すべての汎用レジスタは 32 ビットで，メモリのアクセスは 64 ビット単位になっており高速性を追及しています．

PDP-11 はどこへ(互換性をどうしたか)

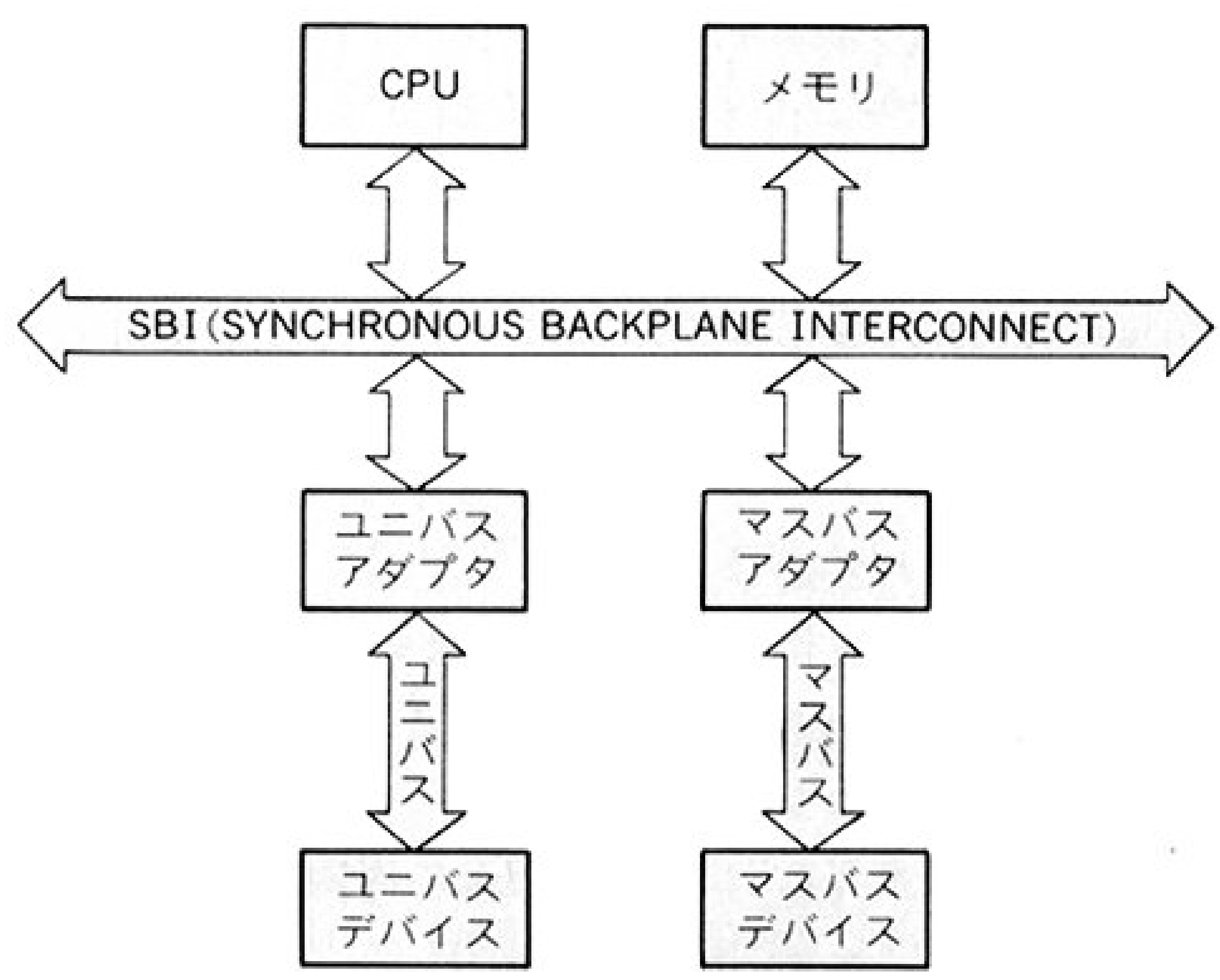
VAX を開発するにあたり問題となったのが，PDP-11 の多くのアプリケーションをどのようにするかということでした．これは簡単にはいきませんが，PDP-11 の互換モードで解決しています．

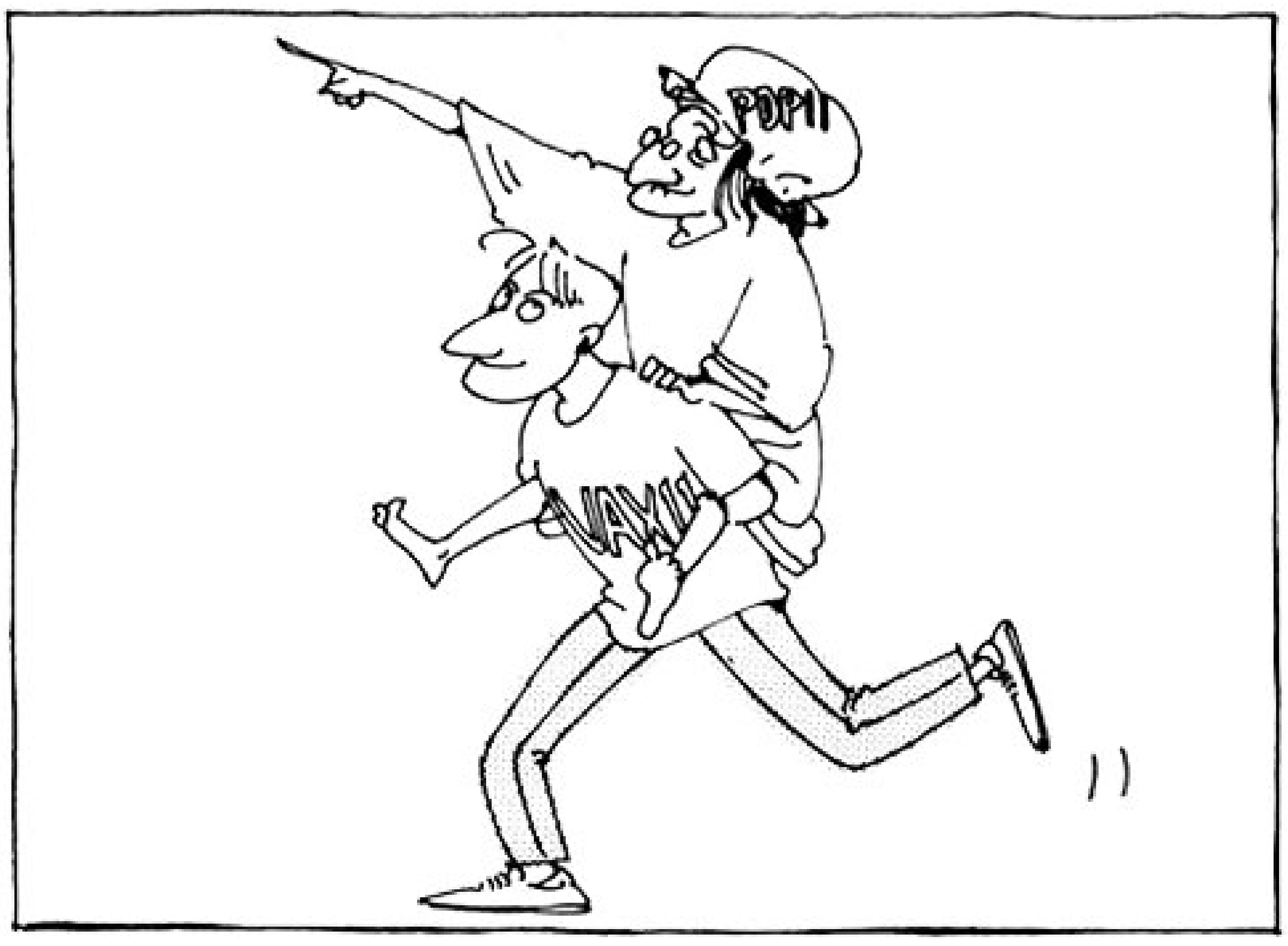
VAX11 の 11 が PDP-11 互換性をハードウェアで有していることを意味しています．このことにより，アプリケーションをわずかな変更で簡単に互換モードで走らせることができ，また新しいアプリケーションは VAX ネイティブモードで開発するといったように，非常にスムースに PDP-11 から VAX11 への移行ができました．

〈写真 5〉
VAX11/780



〈図17〉 VAX11/780 のブロック図





数年もすると，ほとんどのアプリケーションがVAX11に移行できたため，VAXから互換性を意味する11という数字がはずされたシステムが提供されるようになりました．

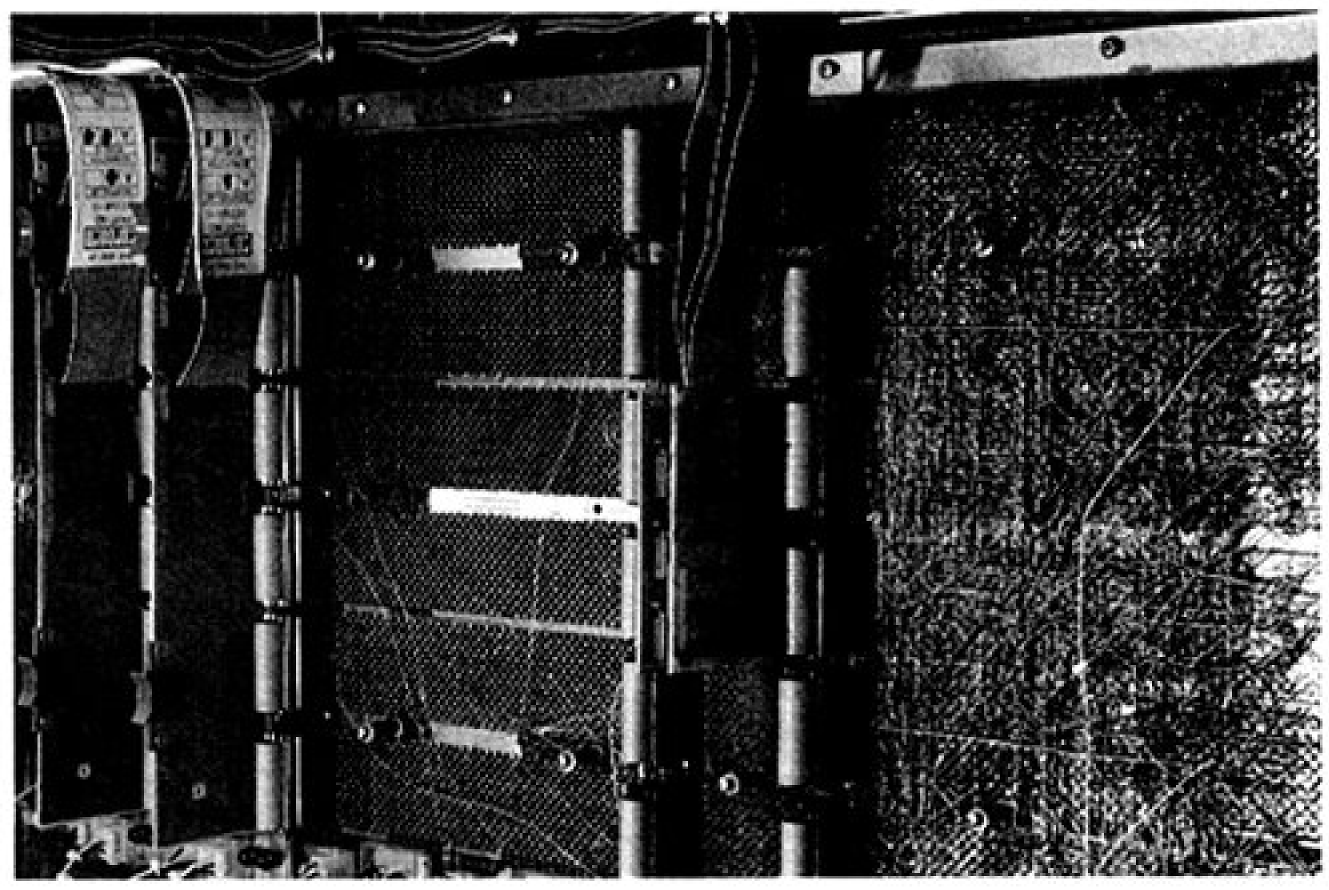
それでは，今までPDP-11のユニバスやマスバス用に開発された周辺機器はどうなったのでしょうか．これもユニバス・アダプタやマスバス・アダプタを付けることで使用可能になっています．

VAX11/780はシステムコンソールとしてLSI11と呼ばれるマイクロコンピュータを持ち，いろいろな診断機能を肩代わりさせていました．

VAX11/780は，非常に高速な13.3Mバイト/秒で動作するSBI(Synchronous Backplane Interconnect)と呼ばれる同期式バスを採用しました(**写真6**)．

SBIにつながるCPU，メモリ，各種コントローラはNEXUS(ネクサス)と呼ばれ，すべてがコンピュータのように独立して動作するようになっています．

SBIバスは形を変え1チップ化されBIバスとなり，後の高性能VAX8200のバスとして使われていきます．1986年のことです．



〈写真6〉SBIの外観

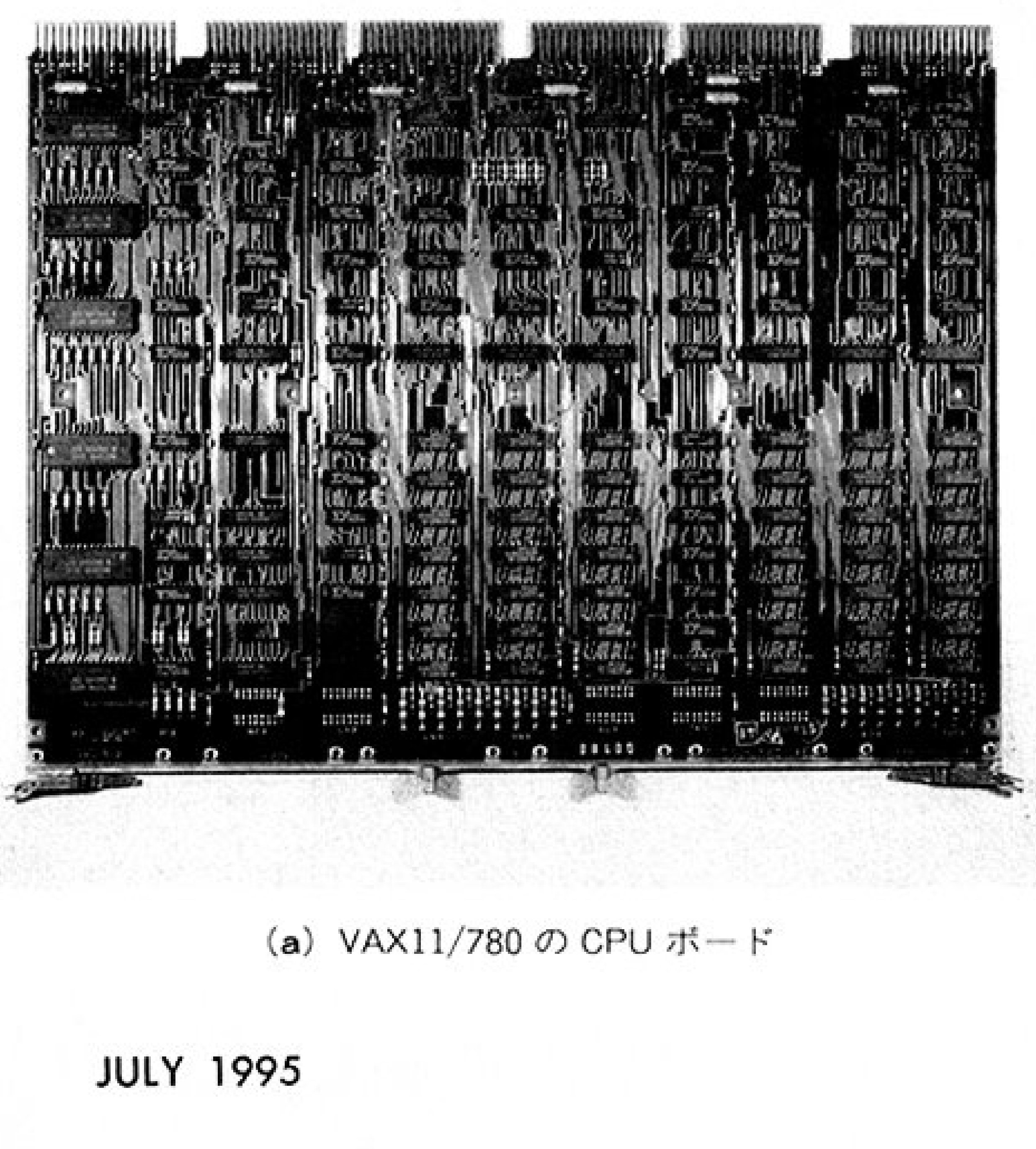


なんともう17年も経ってしまったのかと思うと，いろいろな思い出が走馬灯のように僕の心の中を駆け巡りました．

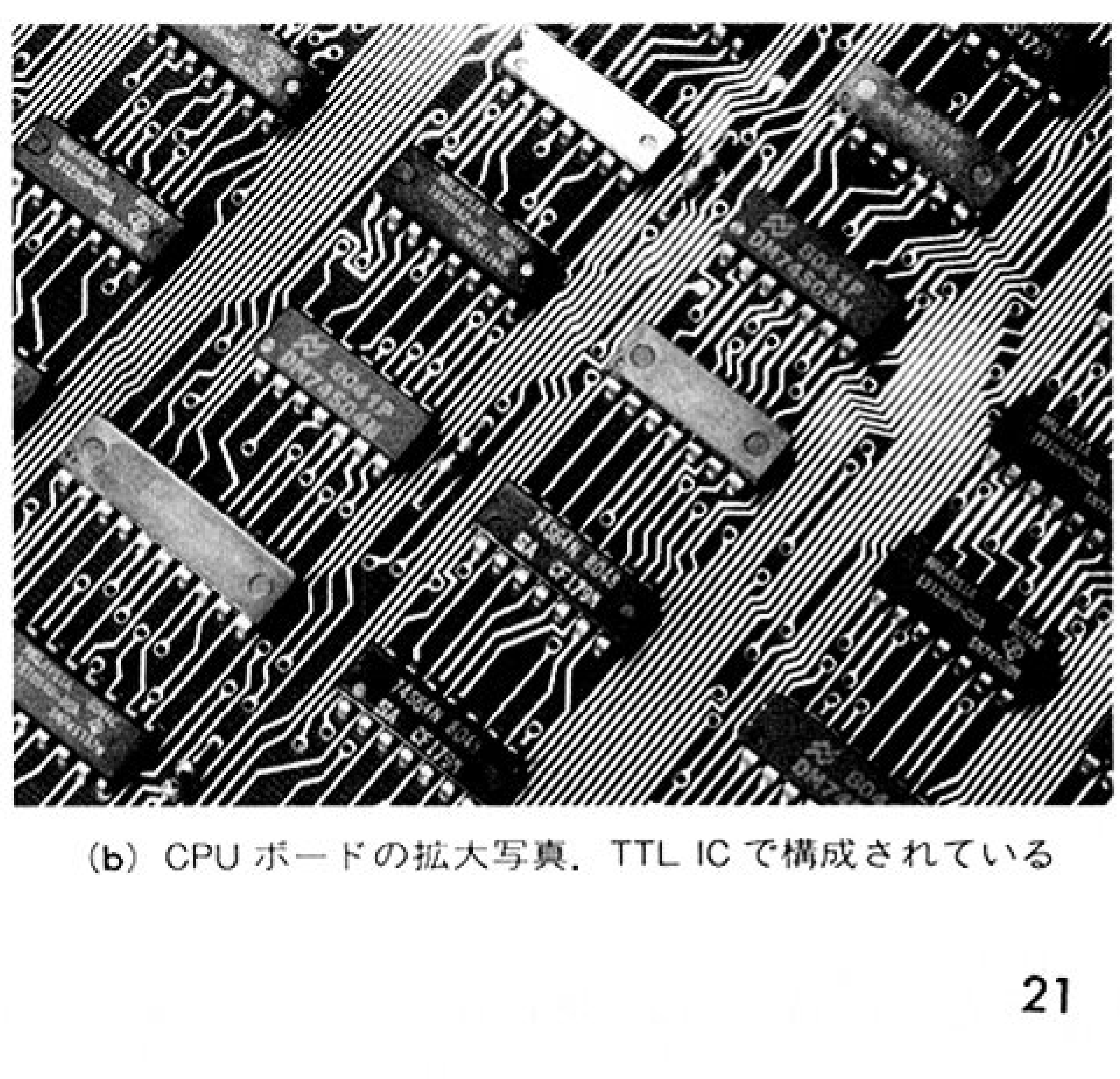
なんと百数十台の端末をサービスした

1977年から1984年まで，VAX11/780が市場に数多く出荷されました．

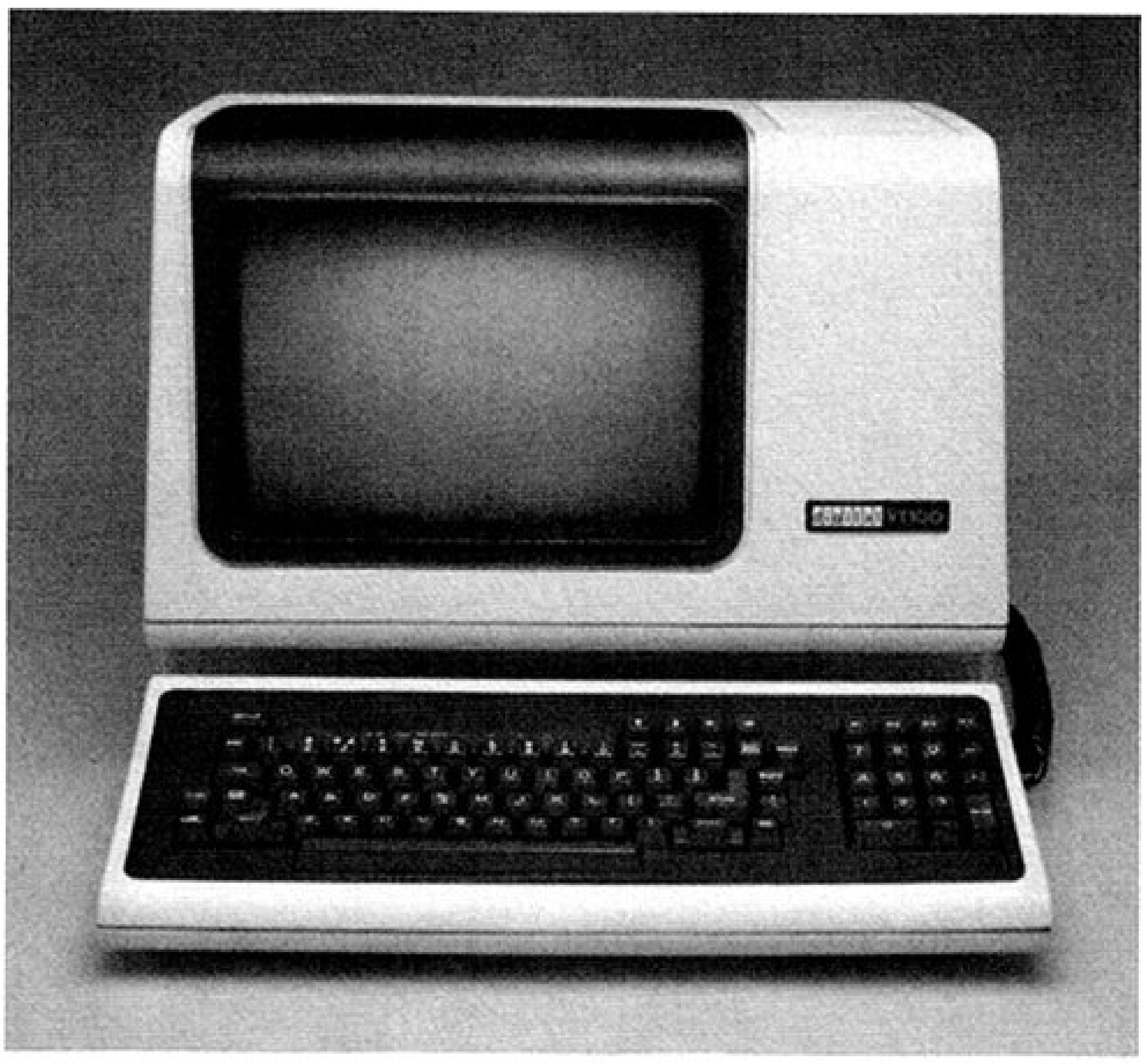
〈写真7〉VAX11/780のCPUボード



(a) VAX11/780のCPUボード



(b) CPUボードの拡大写真．TTL ICで構成されている



〈写真 8〉 VT100

VAX11/780 の主記憶は最低 512 K バイトから最大 16 M バイトまで，キャッシュメモリは 8 K バイト，CPU は縦横 40 センチ，30 センチくらいの大きさのモジュール 20 枚で構成されました(**写真 7**)．

メモリやその他のコントローラを含めると縦 1.8 メートル，横 2 メートル，奥行き 1 メートルほどのキャビネットに納められ，1 MIPS の性能をもつ，当時としては最高のスーパーミニコンでした．

外部記憶装置は数十 M バイトから数百 M バイトの取り外しが可能なものでした(重さ 10 数キログラム程度あり，直径 30 cm の円盤を 10 枚くらい重ねたものでリムーバブルパックと呼ばれていた)．

そしてなんと，端末を百数十台も接続してそれなりのリアルタイム性能でエディタやアプリケーションなどを使用していました．

〈表 2〉 VT100 の文字(VT100 のカタログから引用)

VT100標準文字

VT100で表示可能な標準文字は次の通りです。

Octal Code	Char	Octal Code	Char
040	SP	120	P
041	!	121	Q
042	"	122	R
043	#	123	S
044	$	124	T
045	%	125	U
046	&	126	V
047	'	127	W
050	(	130	X
051	)	131	Y
052	*	132	Z
053	+	133	[
054	,	134	＼
055	—	135	]
056	.	136	∧
057	/	137	—
060	0	140	'
061	1	141	a
062	2	142	b
063	3	143	c
064	4	144	d
065	5	145	e
066	6	146	f
067	7	147	g
070	8	150	h
071	9	151	i
072	:	152	j
073	;	153	k
074	<	154	l
075	=	155	m
076	>	156	n
077	?	157	o
100	@	160	p
101	A	161	q
102	B	162	r
103	C	163	s
104	D	164	t
105	E	165	u
106	F	166	v
107	G	167	w
110	H	170	x
111	I	171	y
112	J	172	z
113	K	173	{
114	L	174	|
115	M	175	}
116	N	176	~
117	O		

スペシャル グラフィック文字

スペシャル グラフィック 文字セットの選択により次のようなグラフィック文字を表示できます。

Octal Code	Special Graphic Char	
137		ブランク
140	◆	ダイヤモンド
141	⁙	チェッカーボード（エラー表示）
142	HT	水平タブ
143	FF	ページ換え
144	CR	キャリッジリターン
145	LF	ラインフィード
146	°	摂氏度シンボル
147	±	プラス／マイナス
150	NL	行換え
151	VT	垂直タブ
152	┘	右下括弧
153	┐	右上括弧
154	┌	左上括弧
155	└	左下括弧
156	┼	クロスライン
157	—	横線－1
160	—	横線－3
161	—	横線－5
162	—	横線－7
163	—	横線－9
164	├	左 "T"
165	┤	右 "T"
166	⊥	底 "T"
167	T	上 "T"
170	│	縦線
171	≦	同等もしくは小
172	≧	同等もしくは大
173	π	円周率
174	≠	不等号
175	£	ポンド・サイン
176	·	中黒点

CTRL(コントロール)キー

CTRLキーは、他のキーと組合わせて押下することにより、コントロールコードを発生します。

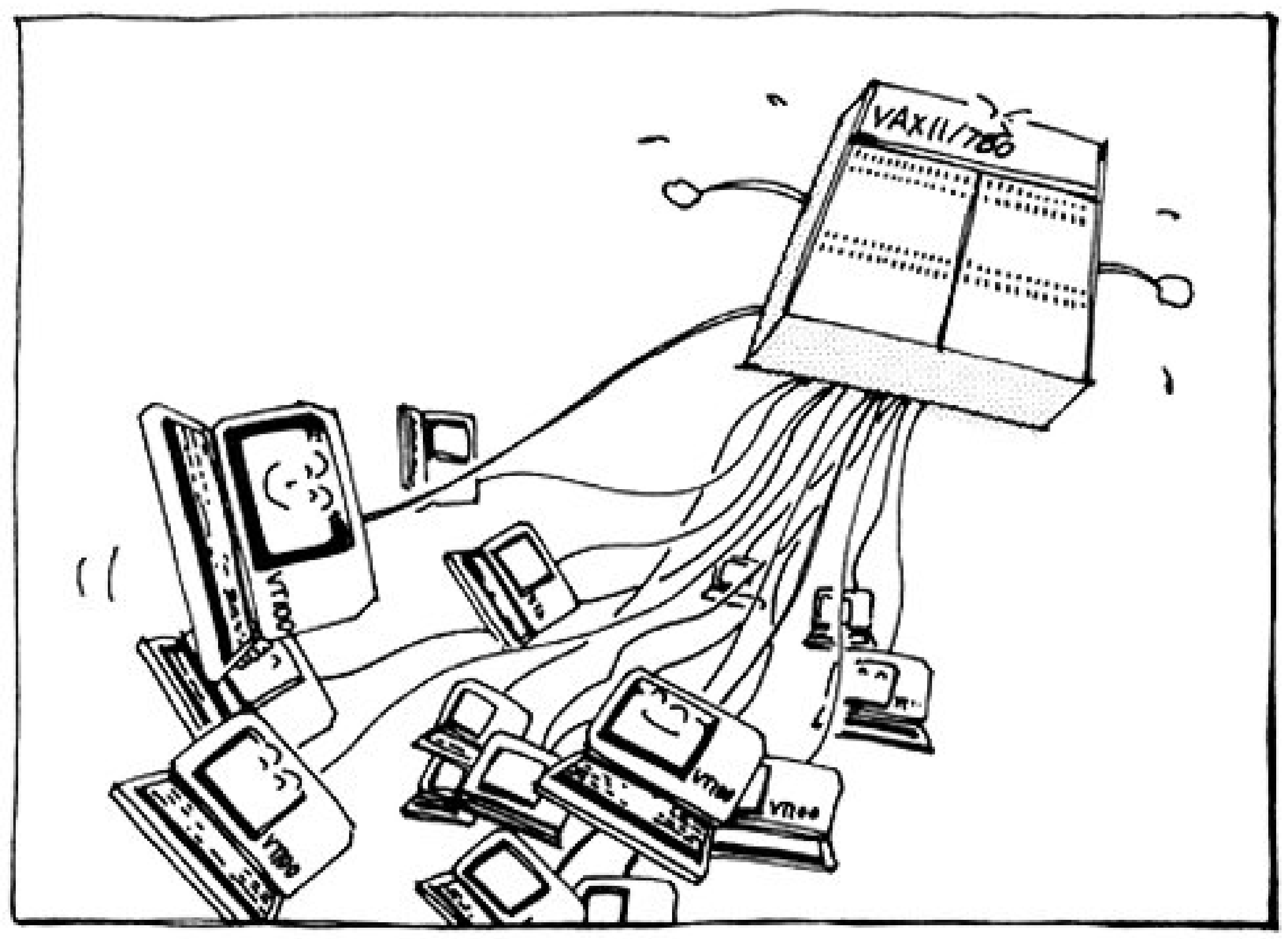
組合わせるキーと、発生するコードの対応は、次のとおりです。

CTRLと組合わされるキー	発生するコントロールコード(オクタル)	ニーモニック
Space Bar	000	NUL
A	001	SOH
B	002	STX
C	003	ETX
D	004	EOT
E	005	ENQ
F	006	ACK
G	007	BEL
H	010	BS
I	011	HT
J	012	LF
K	013	VT
L	014	FF
M	015	CR
N	016	SO
O	017	SI
P	020	DLE
Q	021	DC 1
R	022	DC 2
S	023	DC 3
T	024	DC 4
U	025	NAK
V	026	SYN
W	027	ETB
X	030	CAN
Y	031	EM
Z	032	SUB
[	033	ESC
＼	034	FS
]	035	GS
~	036	RS
?	037	US

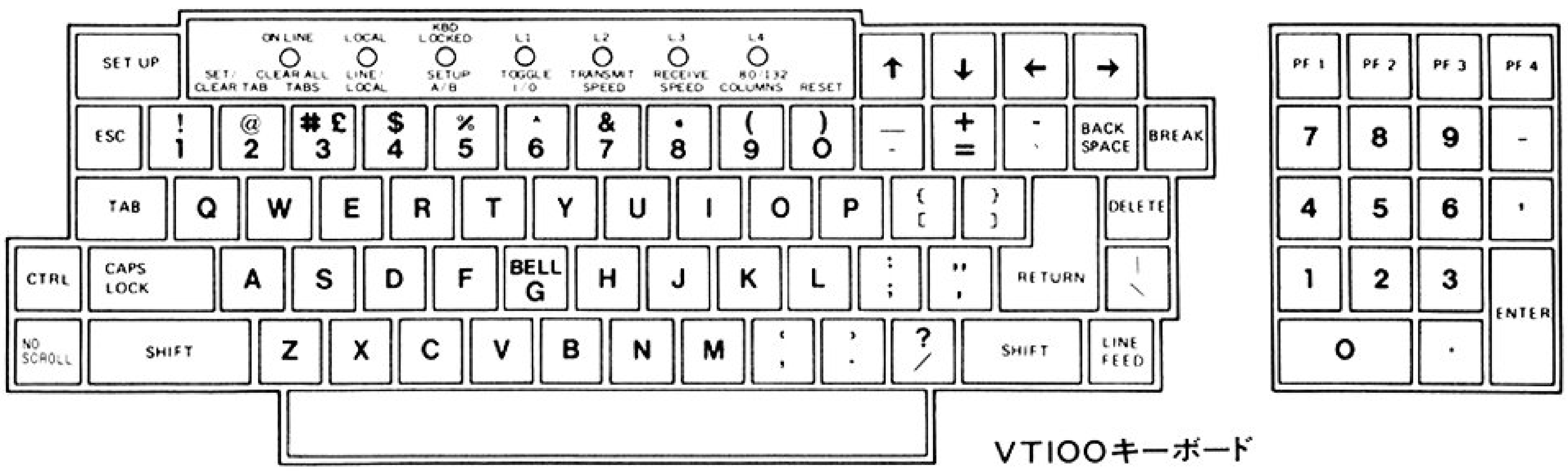
これには，ユニバス上にインテリジェントなコミュニケーションコントローラを複数台つなぎ，VAXの負荷を軽くして端末の制御を行いました．

また，この時代に非常によく用いられたビデオ端末がVT100と呼ばれる有名な製品で，現在でも世の中のいろいろなターミナルアプリケーションの標準ともなっています(**写真8**)．

サードベンダーからパソコンのモニタ上で動作するVT100エミュレータやVT互換端末なども発売されています．VT100のキーボードレイアウトを**図17**に，文字を**表2**に，仕様を**表3**に示します．



〈図18〉VT100のキーボードレイアウト(VT100のカタログから引用)



〈表3〉VT100の仕様(VT100のカタログから引用)

VT100　仕様概要

寸法		高さ	モニタ部　36.8cm　キーボード部　8.9cm
		横幅	モニタ部　45.7cm　キーボード部　45.7cm
		奥行	モニタ部　36.2cm　キーボード部　20.3cm
重量			モニタ部　13.6kg　キーボード部　2.0kg
環境条件	相対温度	動作時	10°〜40℃
		非動作時	−40°〜66℃
	相対湿度	動作時	10〜90%
		非動作時	0〜95%
電源条件		電源電圧	90〜128VAC，180〜256VAC(スイッチ選択)
		電源周波数	50，60Hzいずれにても可
消費電力			最大150W
表示		画面構成	80カラム×24行または123カラム×14行(オプションで24行)
		文字マトリックス	7×9ドット
		キャラクタサイズ	3.35mm×2.0mm(80カラム時)　3.35mm×1.3mm(132カラム時)
		表示画面サイズ	202mm×115mm
		キャラクタセット	ASCII 128文字+32グラフィックキャラクタ
キーボード		配列	タイプライタ型
		補助キーパッド	18キー数字パッド(ピリオド，コンマ，ENTER，ユーザ定義キー付)
		インディケータ	7個のLED表示(ONLINE，LOCAL，KBDLOCKED+4個のユーザ定義LED)
		その他	キー・クリック，ベルコードの発生
送受信		方式	EIA標準
		速度	全二重転送 50，75，110，134.5，150，200，300，600，1200，1800，2000，2400，3600，4800，9600，19200ボー
		コード・コード長	ASCIIコード，7ビット(8ビットの場合，8ビット目はスペース)
		パリティ	キーボード選択による偶数パリティ，奇数パリティ，パリティ無
		同期	キーボード選択によるX−ON，X−OFF制御可能

● MIPSの起源

この頃になるとVAXのメモリや周辺機器の互換機メーカーが多数出現しました.

皆さんはVUPS(VAX UnitPerSecond)という言葉を耳にしたことがありますか? コンピュータの性能比較を行う書物には必ず出てくる名称です.

これは, システムの性能をVAX11/780を1として計るときに用いられます. また, VAX11/780は約1 MIPS(Million Instruction Per Second)とされていました.

現在では, パソコンやワークステーションの性能を計る単位でSPECint/SPECfpなどが採用されていますが, これらの単位もVAXを基準としています.

VAXもまたPDP11と同じように小型化の道をたどります. そしてながーい繁栄がDECに訪れます.

マイクロプロセッサとパソコンの台頭

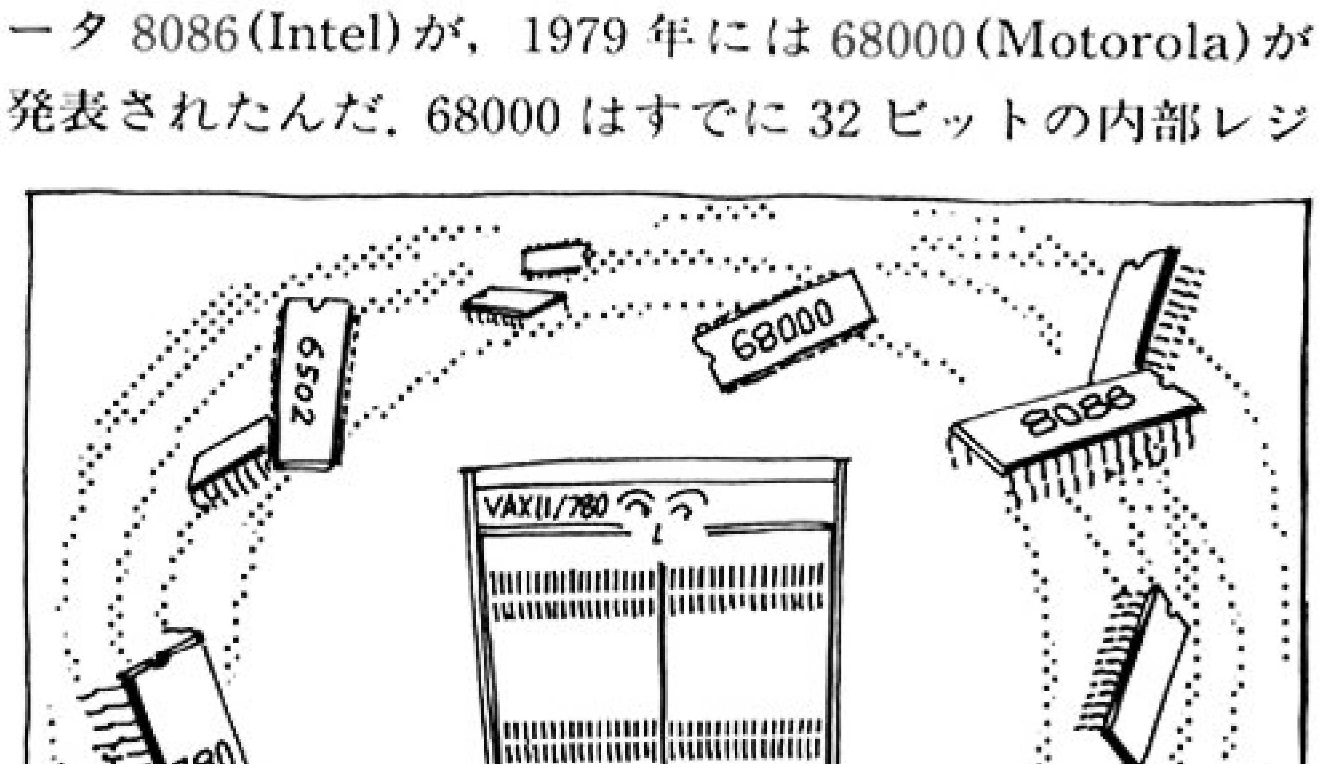
● マイクロプロセッサの誕生

1988年の非常に寒いある日, ケンがマイクロプロセッサについてどう思うか, PCの性能は, 価格は, などと聞いてきた. それについて僕はいろいろなメーカーのものを使っていたので即座に答えた.

勝男 「そうだね, まず初期の頃からどうだったのかを見てみよう. まず, PDP-11が出た1970年にはケンも知ってるとおり, マイクロプロセッサなんてまだなかったよ.

世界初の4ビットマイクロプロセッサ4004(Intel)が出たのは1971年で, 1972年には8ビットの8008, 1973年に8080, 1974年に6800(Motorola)が出て, 1975年には6502(MOS Technology. 後にAppleIIに採用された), 1976年にはZ80(Zilog), 8085などが出てきたよ.

まだPDP-11が全盛の時さ. そして1977年にVAXが出て, その翌年にやっと16ビットマイクロコンピュータ8086(Intel)が, 1979年には68000(Motorola)が発表されたんだ. 68000はすでに32ビットの内部レジスタを持っていてびっくりしたよ.



1982年に80286, 1983年に68020, もうその後の発展はケンも知ってるとおりだよ. でも当初8086はアドレスにセグメントを使っていて, すごくいびつでいい印象はなかったな. 将来は68000系が主流だと思ってたんだが, IBMがIntelの8088を採用したためパソコンの主流になれなかったんだ.

● パソコンの誕生

1970年代も終わりになると, パソコンやマイコンということばが使われ始めたんだ.

ケン, なんとAppleIIというパソコンがVAXの生まれた1977年に出たんだ. 8ビットMPUで6502を使い, メモリは48Kバイト, 143Kバイトの5インチ片面フロッピディスク・ドライブ, 白黒モニタを付けて100万円(本体価格は約40万円)もしてたんだ.

でも, フロッピディスクはもう素晴らしいスピードでデータをアクセスしたから一度使うともう病み付きになったよ.

そういえば, その頃はディスクも値段が高いんで, 書き込み許可用の切り込みが右上に5ミリほどの大きさで入ってるんですが, それと同じものを左側に切ってやると, なんと片面のディスケットが両面とも使用可能になるんですよ.

それを当て込んだ業者が, 「切り込み専用カッター」なんてものを結構な値段で売り出してたなぁ. 金のない僕なんかハサミを使って切り込みを入れたものですよ.

その当時のロードデバイスはカセットテープ, 例の音楽を聞くのと同じものがプログラムやデータの保存や読み込み用のデバイスとして使われてて, フロッピディスクを個人で使うなんてもう贅沢の最たるものでしたよ. ところでケンはパソコンは使わないの?」

ケン 「もっと高性能なら使ってもいいが, 私としては, ビジネスとして成り立つか, 利益をどうあげるかという問題のほうに興味があるね. もしうちが参入するとしてもまだまだ先になると思うよ」

勝男 「ふーん. 僕のApple上ではMicrosoft製Z80カードを付けてCPM56というOS(Digital Research)が動作してるんだよ.

これがすごくって, すでに表計算ソフトのVisicalc, BASIC, FORTH, Fortran, COBOL, LISP, Small-Talk, それにChoplifterっていう戦闘ゲームなど, ありとあらゆるソフトが走っていて, それらを買うだけでも小使いが吹っとんでしまいますよ.

僕がまだ『パソコン青年』と呼ばれている頃でしたね.

これから数年後, 日本ではPC9801と呼ばれるパソコンが登場します.

ちなみにわが家では, AppleII, PC9801F2, Mac512

など昔のマシンが今でもすこぶる快適？ に動作していますよ．ケン，一度見にきませんか？ それを見たら，必ずこれは商売になるとケンも納得しますよ」

ではということで，ケンは友人4人を連れてくることになった．

当日私は4人を見て驚きました．例の会社の4人ではないか．

イブム・アムダンに「どう仕事は」と聞くと，彼は今は夢を実現してコンピュータを製造してるそうで，IBMの大型互換機で今はフュージット・フェイカムと一緒にやってるということだった．

ニキーデ・ン・ニークは儲けた金で，僕の使っているPCを作ってるとのこと．

フュージット・フェイカムやイターキ・イタークも超大型からPCまで幅広く手掛けているそうだが，PCではニークがやり手で困っているとのことだった．

でも，彼らを見ていると様々な経験を積みながら夢を実現したなんて素晴らしい人達だなと思った．

今まで彼らの名前にある種の親しみを持っていたので綴りを教えてもらって非常に驚くと同時に納得した．なんと，

イブム・アムダンはIBM・AMDAHL
フュージット・フェイカムはFujitsu・FACOM
イターキ・イタークはHITACHI・HITAC
ニキーデ・ン・ニークはNICHIDE・N・NEC

とのこと．

私のコンピュータ人生にかかわるこの人達に感謝．

ケンも彼らを見て，DECもパソコンやるかと目を輝かせていた．

これが現実のものになるのは，この後さらに数年の歳月が必要だった．そして時は瞬く間に過ぎ，DECも僕も非常にいろいろな経験を積んでいくことになる．

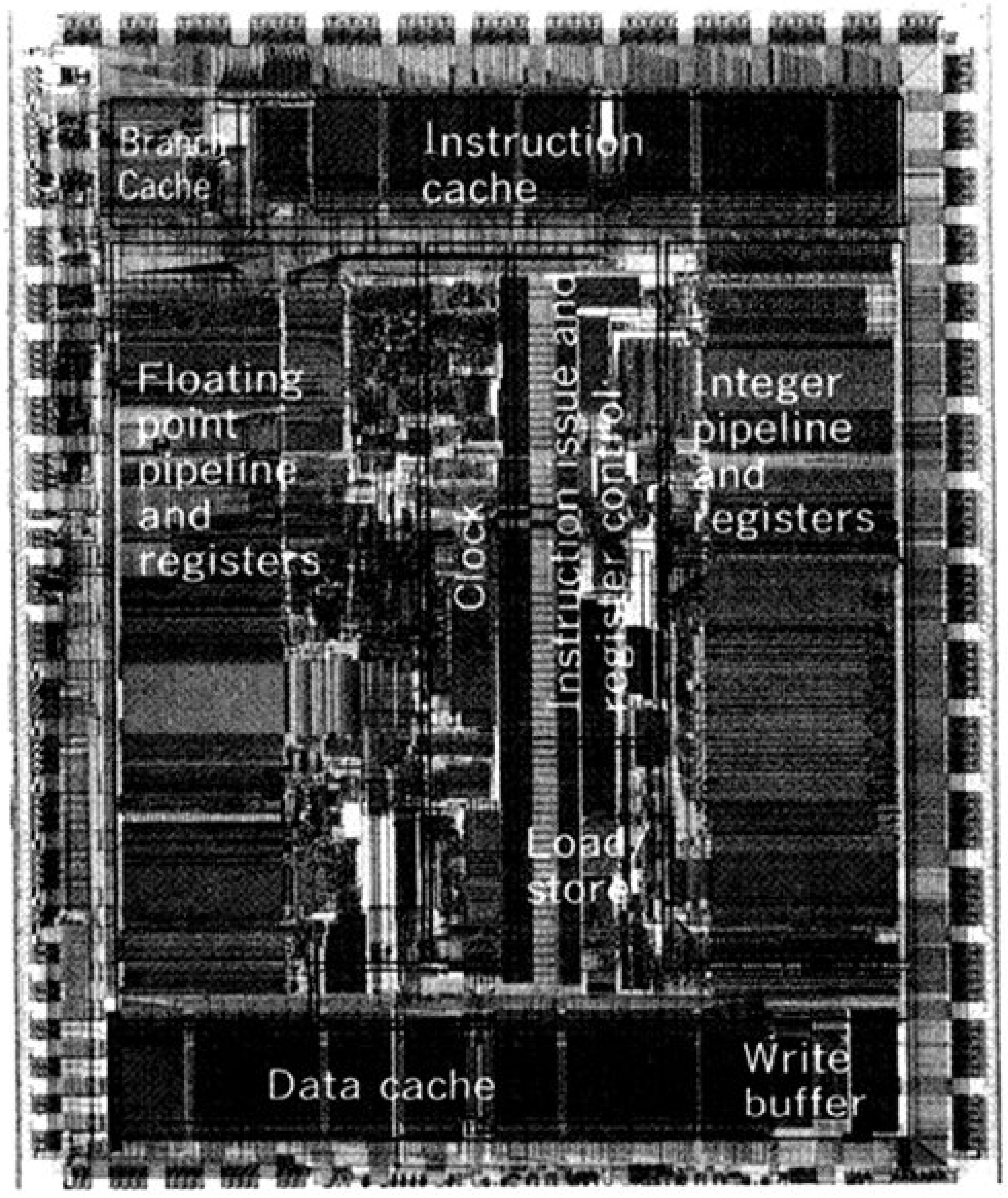
そして僕は，久しぶりに日本に帰ることになった．時はすでに1995年の初めのことです．

● 32ビットから64ビットへ，Alphaチップの到来

日本での生活も落ち着きを取り戻しつつあったある日，アメリカにいた時からずーとファンで，僕の技術の宝庫であったトランジット技術から，Alphaチップについて記事を書いてくれないかと依頼が来た．そこで1週間で約30ページも書いたら，担当者が各社1ページずつでいいんですよ，だって．

僕は気を取り直して，では後日50ページくらいの「DECの主張，初心者向けAlphaと他社RISCアーキテクチャ徹底比較特集」をやりましょうと言ったんだが聞いてもらえるかな？

そして，30ページを1/2ページに圧縮したAlpha解説記事ができ上がった．



〈写真9〉 Alpha 21064

DECはPDPからVAX発売以来1990年頃までミニコンメーカーとして知られていました．

また，VAXは32ビットスーパミニコンのベストセラーとして有名でした．VAXアーキテクチャはVAXtationと呼ばれるワークステーションにも採用され同じVMSが動作します．

CISC型コンピュータのVAXとして，これ以上の性能向上には様々な困難がつきまとうため，DECはアーキテクチャを一新した64ビットのRISC型チップAlpha(21064)を1992年9月に発表しました(**写真9**)．

このチップは200 MHzで動作，170万トランジスタを集積，8Kバイトの1次命令キャッシュ，8Kバイトの1次データキャッシュを内蔵，外部データバス128ビットで2命令同時実行が可能な高性能なチップです．

できるだけ不要な回路は省き，周波数を上げて最高のパフォーマンスが得られるようにし，スーパースケーラ/スーパーパイプラインを採用しました．

ソフトウェア互換性を保持した21164は，300 MHzで動作，930万トランジスタを集積，96Kバイトの2次キャッシュまでも内蔵，4命令同時実行が可能なチップです．330 SPECint，500 SPECfp，1200 MIPS，600 MFLOPSという性能です．

第2章

IBM PC+MS-DOSは どのようにして標準となりえたか

世界標準パソコンIBM PCと MS-DOSの変遷

Macorney E. Saitmayer

● プロローグ

1980年の初めまでは，コンピュータメーカー各社が独自のアーキテクチャに基づいてその先進性を競っていた時代であり，いろいろなハードウェア構成や基本ソフト(OS)が市場に乱立していました．

しかし，IBMがIBM PCを発表した時を境に，欧米のパソコン市場は一気にIBM PC(とその互換機)，そしてMacintoshだけになってしまい，4,5年前からは単に「PC(ピーシー)」と呼ぶときにはIBM PC/AT互換機のことを指すようになりました．

特にPCの進化はCPUの進歩もさることながら，数十個のICで構成されていた周辺回路を1チップにした「チップセット」と呼ばれるLSIの集積度の向上や，Windows 3.1の登場とともに加速的に飛躍したビデオアダプタ，そして大容量化したハードディスクなどが追い風となっています．

常に成長し続けるパソコン市場に，大企業からベンチャー企業まで，ありとあらゆる規模の会社がビジネスチャンスを狙いながら最新鋭の技術を投入してきた結果がそれであり，今もなおその進化は止まっていません．

ここでは，そのIBM PCの歴史を振り返りながら，そのアーキテクチャやハードウェアの進化を追っていきたいと思います．

なお，以下の文章ではあえてことわりのない限り，IBM PCを単に「PC」と表記することにし，IBM PC/ATなどもPC/ATと呼ぶことにします．

初代PCから今のIBM PCまで

ここでは，**表1**の年表にしたがって，PCの歴史をたどってみることにします．

初代PC(写真1)

● 初代PCのスペック

あえて，「初代」と付けたのは，その昔純粋に「IBM PC」と呼ばれていた機種が存在していたからで，ウルトラマンファミリの初代ウルトラマンと同様の意味付けでそう呼ぶことにしました．1981年に登場した初代PCのスペックを**表2**に示します．



〈写真1〉 初代PC(IBM PC)

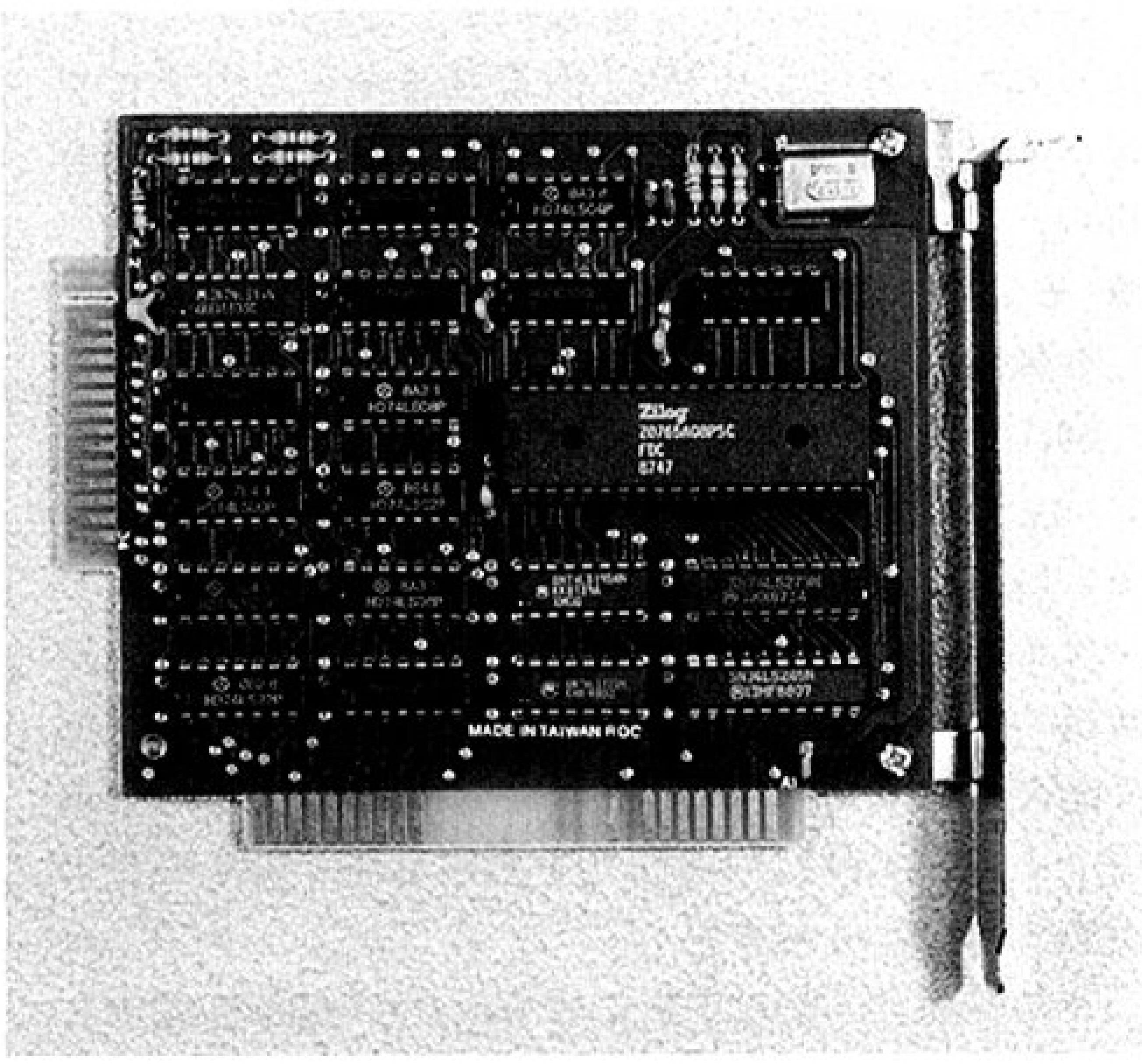
マザーボード上には**表2**のほかに数十個のTTL ICが搭載されており，フロッピディスクドライブ・インターフェースカード(**写真2**)，ハードディスクドライブ・インターフェースカード，ディスプレイカード，シリアル/パラレルカードをスロットに装着するように設計されていました．

特に，ディスプレイカードは初代PC登場時は，MDP(Motochrome Display and Printer adapter)とCGA(Color Graphics Adapter)があり，BIOS ROMにそれらのためのドライバコード(いわゆるビデオBIOS)がプログラムされていました(**写真3**)．

また，この頃はフロッピディスクの容量も160 Kバイトしかなく，初代PCはカセットテープを補助記憶として利用していたのでオーディオインターフェースを持っていました．フロッピディスク・ベースのPC-DOS(IBMから供給されるMS-DOS)のバージョンは1.0を使用していました．

〈表1〉
IBM Personal Computer(PC)の歴史年表

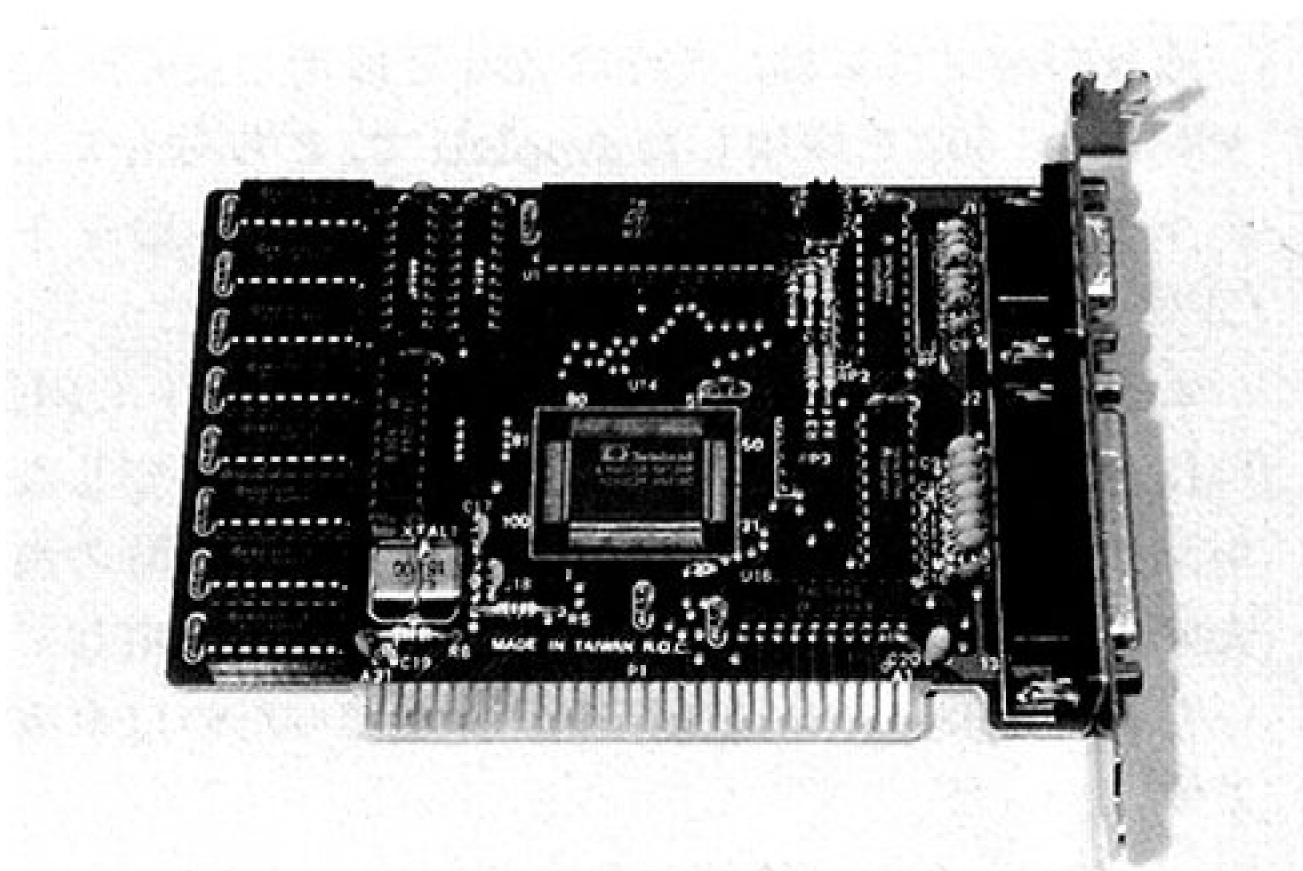
年代	Intel/IBM	Microsoft
1978	8086 プロセッサが登場する	
1979	8088 プロセッサが登場する	
1981	8月　初代 IBM PC が IBM から発売される(4.77 MHz 8088)	MS-DOS 1.0
1982	80286 プロセッサ(16 M バイトアドレス空間)が登場する	
1983	3月　PC/XT が登場する	MS-DOS 2.0
1984	6月　Compaq System Pro(8086 搭載)が発売される 8月　PC/AT(6 MHz 80286)が登場する	MS-DOS 2.11, 3.0, 3.1
1985	80386DX プロセッサが登場する(32 ビットデータ，4 G バイトアドレス空間)	
1986	4月　PC/AT の 8 MHz 版が発売される	MS-DOS 3.2
1987	7月　IBM が PC/AT の販売を終了する 同　PS/2 シリーズが IBM より発売開始される	
1988	80386SX プロセッサが登場する	MS-DOS 4.0
1989	i486DX プロセッサが登場する	
1991	80386 搭載の PC/AT 互換機が発売される i486SX プロセッサが登場する マルチメディア志向のプログラム開発が増加してくる	MS-DOS 5.0
1992	i486DX2(2倍クロック)プロセッサが登場する ローカルバス：競合する二つの規格(VL と PCI)が市場に普及してくる	
1993	i486 の後継プロセッサとして Pentium が登場する Intel DX4(3倍クロック)プロセッサが登場する 16 M ビットメモリの出荷が始まる	
1994	Pentium の高速版が登場する Intel と競合する数社が Pentium/486 互換プロセッサを発表/出荷する	MS-DOS 6.0
1995	Pentium の後継プロセッサとして P6(コードネーム)が登場する IBM が本格的に IBM PC 市場に復帰する	



〈写真2〉初期のフロッピディスクドライブ・インターフェースカード

〈表2〉初代 PC のスペック

CPU	i8088-4.77 MHz
タイマー	i8253
DMA コントローラ	i8237A
割り込みコントローラ	i8259A
パラレル I/O	i8255A
標準メモリ	16 K バイト
スロット	8 ビット拡張スロット
フロッピディスク	160 K バイト
その他	クロックジェネレータ, BIOS 用 ROM



〈写真3〉MDP/CGA 互換カード

● PC アーキテクチャの発展

当時の米国のパソコン市場は，AppleII や Z80 を使用した CP/M マシンが主流であり，8 ビット・データバスでありながら 16 ビットアーキテクチャをもつ 8088 を搭載した初代 PC の登場に，他の競合メーカーは市場規模の拡大につながるとして IBM の市場への参入を歓迎していました．

しかし，それらの企業は，その後ハード/ソフトの両面において 16 ビットへの移行でつまずいたり，市場の読みを誤って次々と淘汰されていくことになります．

結果的に，この初代 PC を出発点とした PC アーキテクチャの発展と PC 市場の爆発的な拡大が Intel/Microsoft 両社の隆盛を決定づけることになります．

● 初代 PC のメモリ構成

初代 PC のメモリ構成は，8086/8088 の 1 M バイトの空間のうち，**00000H～9FFFFH** の 640 K を RAM エリア(実装されているメモリはもっと少なかったが)とし，残りの **A0000H～FFFFFH** を ROM やビデオメモリ空間としていました(図 1)．

当時は，OS もアプリケーションも今と比べれば非常に小規模だったので，「640 K バイトもの大容量」のメモリを必要とすることはありませんでしたし，その空間にメモリチップをすべて実装することは，経済的にも不可能なことでした．

しかしながら，これ以降におけるソフトウェアの大規模化と半導体製造技術の進歩により，当初は十分すぎるほど広いと思われていた 640 K バイトのメモリ空間が逆に「640 K の壁」として大きな影響を後々まで与えることになります．

PC/XT(写真 4)

その後，1983 年に IBM はホビーユースとビジネス

コラム●なぜ，初代 PC に 8088 が採用されたのか

IBM PC の最初の機種に Intel の 8088 が採用され，その後の Intel が開発する CPU が次々と PC に採用されていきました．

その結果として，Intel が半導体業界の巨人となったわけですが，もし初代 PC の CPU が 8088 でなかったら，この業界の勢力分布もまったく違ったものになっていたでしょう．

では，なぜ 8088 だったのか，どうして 8086 ではなかったのか，あるいは 68000 ではだめだったのか，この疑問をここで明らかにしておくことも PC の歴史の上では重要なことでしょう．

さて，初代 PC の CPU の決定のいきさつとして，以下のようなものがありました．

● 他社の PC よりも性能がよくないといけない．しかし，高くてはだめ

当然のことながら，IBM はパソコン市場へ遅れて参入するわけですから，他社のパソコンよりもよい製品を出す必要がありました．

他社のパソコンは，当時は Z80 を採用した CP/M マシンか 6502 を採用した AppleII で，どちらも 8 ビット CPU でしたから，IBM としては 16 ビット CPU を採用したかったわけです．

となると，その当時に入手できる 16 ビット CPU には 8086/8088 か 68000 ぐらいが妥当なものでしたが，68000 の価格は結構高く(何といっても当時の最先端技術である 64 ピン DIP パッケージを採用していた！)，8086(48 ピン)の方が安価だったのは事実です．

● しかし，あまり性能が良すぎてはいけない

当時の IBM のビジネスの主力は何といってもメインフレームで，その周辺機器として販売しているターミナルが結構儲る(ぼれる)商売だったそうです．

もし，IBM PC の性能対価格比が良すぎて，PC 用のターミナルエミュレータ・ソフトが供給されてしまうと，メインフレーム事業部のおいしい部分が根こそぎ消滅してしまう恐れがあり，IBM PC プロジェクトそのものが中止になる可能性があります．

そこで，16 ビットへ移行できる足掛かりを残した形で 8086 よりも性能の劣る 8088 を選んだそうです．

8088 でも，当時の他社のパソコンに比べれば優位に立てますし，社内の顔もつぶさなくてすむため，これが最善の選択だったわけです．

8086 よりもはるかに性能のよい 68000 なんかを使ったら，それこそ IBM の重役達からこのプロジェクトはコテンパンに打ちのめされて，門前払いを食らっていたでしょう．

● そして，IBM の言うことを聞いてくれる会社から供給を受けないといけない

初代 PC の開発当時の Intel は，その頃でもシリコンバレーの主要なベンチャ企業ではあったのですが，今とは比べものにならないほど小規模な会社でした．

また，一方の Motorola のほうは昔も今も親方星条旗の巨大企業であり，68000 はその半導体事業部で開発されたものです(この会社の本業は通信機器で，その製品開発の課程で半導体を作りはじめたのが始まりだそうです)．

その当時の IBM としては Intel の 8088 を採用し，Intel の株を所有することによって強力な発言権を確保しておき，初代 PC 用の 8088 を安定供給させ，なおかつ次期 CPU の開発/供給においても IBM へ優先的に製品供給を行わせたかったようです．

というのが，初代 PC に IBM が Intel の 8088 を採用した理由といわれています．いつの時代でも，どの業界でも，技術的に一番よい製品が一番売れるとは限らないという例にすぎないと言えばそのとおりですが，あの当時に今の PC 市場を予想できた人がもしいたならば，筆者は 68000 を採用して欲しかったと思います．

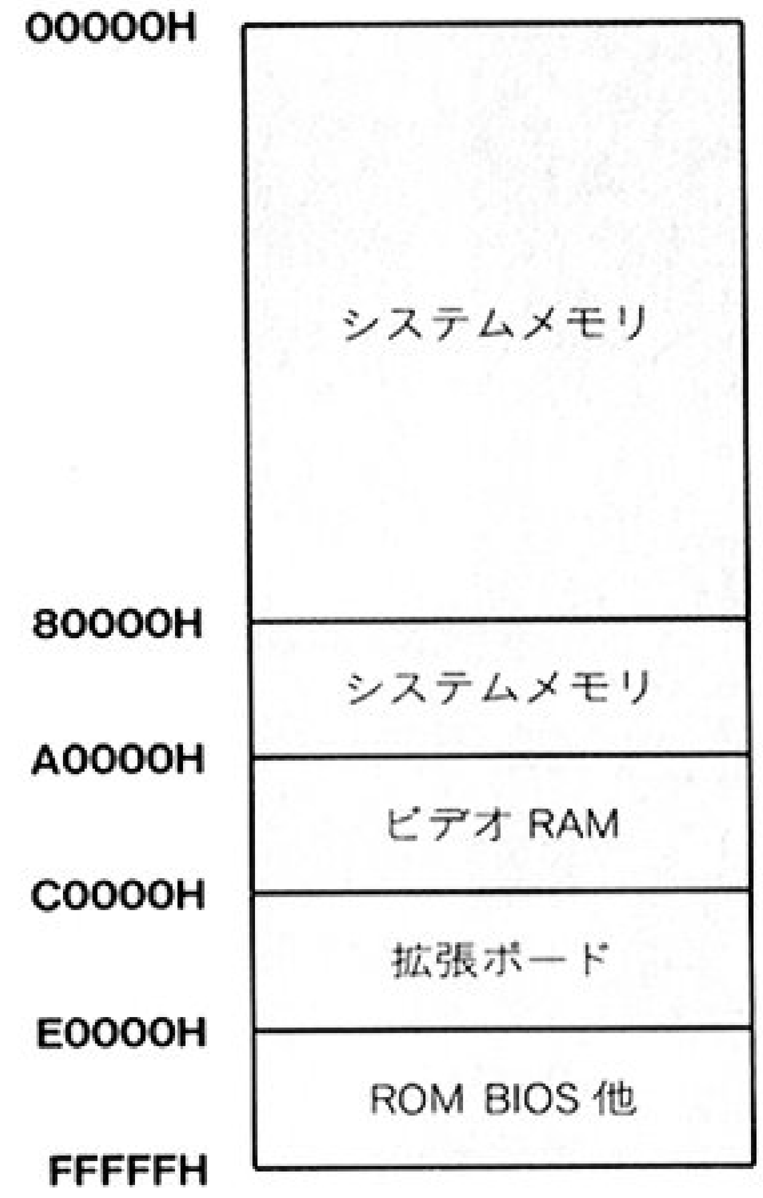
ユースのために特化した2種類の後継機種を発表しました．

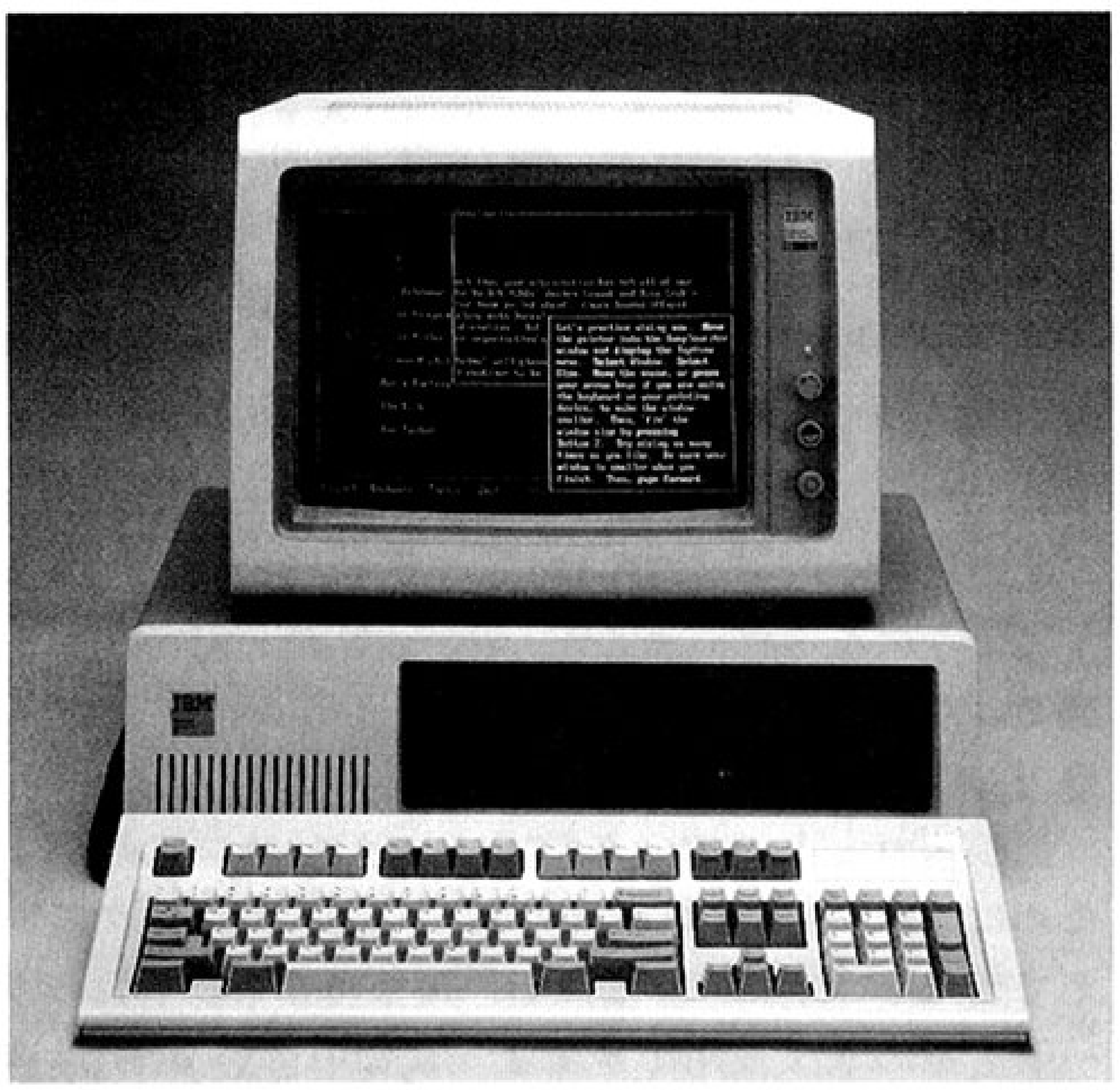
まず，ホビーユース向けには，PCjr(**写真5**)という機種が投入されました．この機種は初代PCの拡張性を大幅に削り，グラフィックボードとジョイスティックを標準搭載した機種でしたが，業界の大きな期待にもかかわらず市場にはまったく受け入れられずに消えていきました．

もう一方のPC/XTは，ビジネス向けに開発された機種でスペックは**表3**のとおりです．ビデオアダプタもPGA(Professional Graphics Adapter)が追加されました．

このPC/XTは，CPUやバスの仕様が初代PCと同一であり，次のPC/ATが登場したときスロットが16ビットになった時点で，この8ビット用のスロットを8ビットバスあるいはXTバスと呼ぶようになりました(**写真6**)．

〈図1〉
初代PCのメモリ構成





〈写真4〉 PC/XT



〈写真5〉 PCjr

PC/AT(写真7)

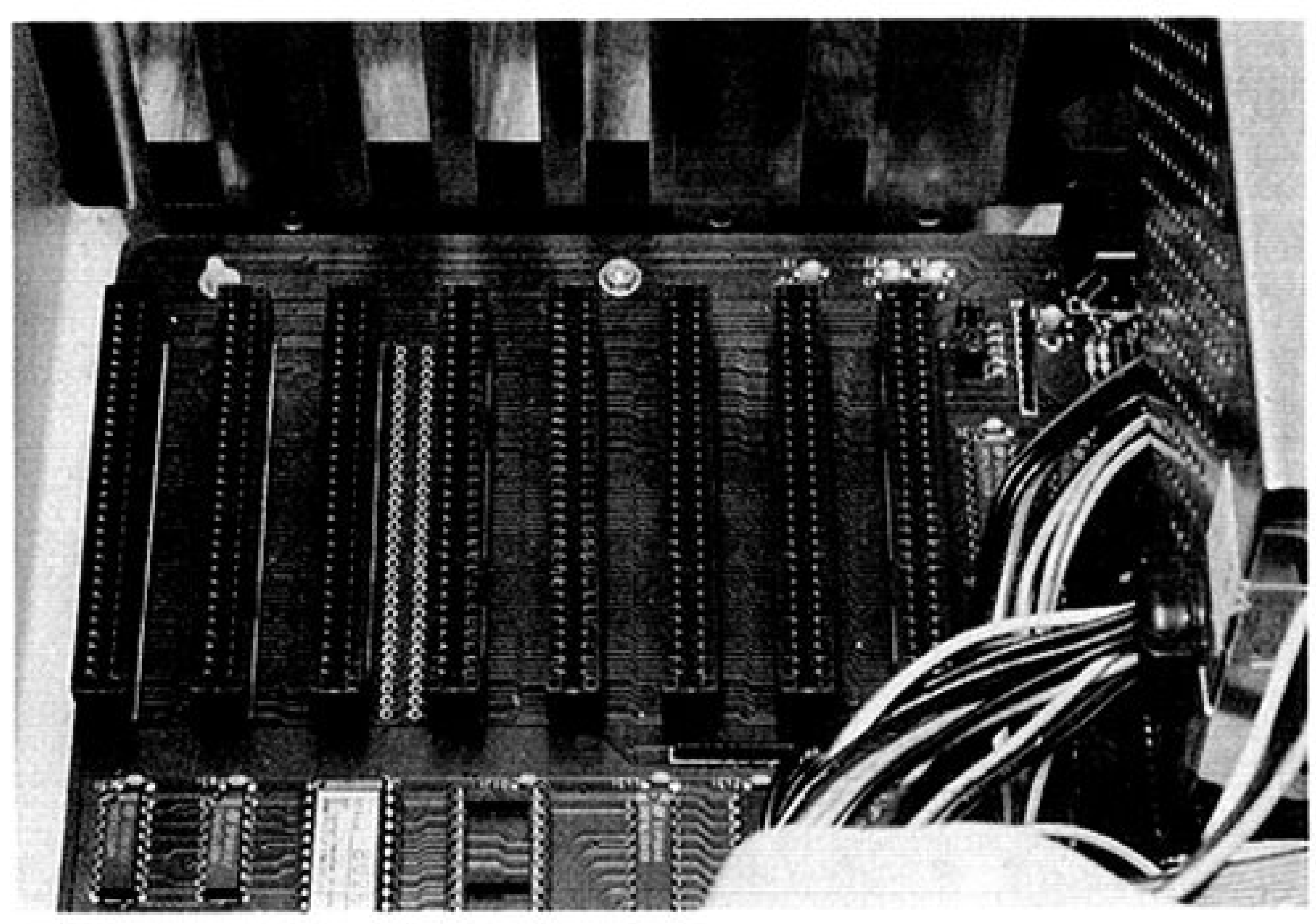
80286の採用

2機種のPC/XTのマイナーチェンジにはさまれる形で，PC/ATが1984年8月にPCの最上位機種として発表されました．

このPC/ATが，その後のPC市場に決定的な方向付けをもたらした機種であり，今のPC互換機市場の

〈表3〉 PC/XTのスペック

CPU	i8088-4.77 MHz
タイマー	i8253
DMAコントローラ	i8237A
割り込みコントローラ	i8259A
パラレルI/O	i8255A
標準メモリ	64Kバイト
スロット	8ビット拡張スロット
フロッピディスク	360Kバイト
ハードディスク	10Mバイト
その他	クロックジェネレータ，BIOS用ROM

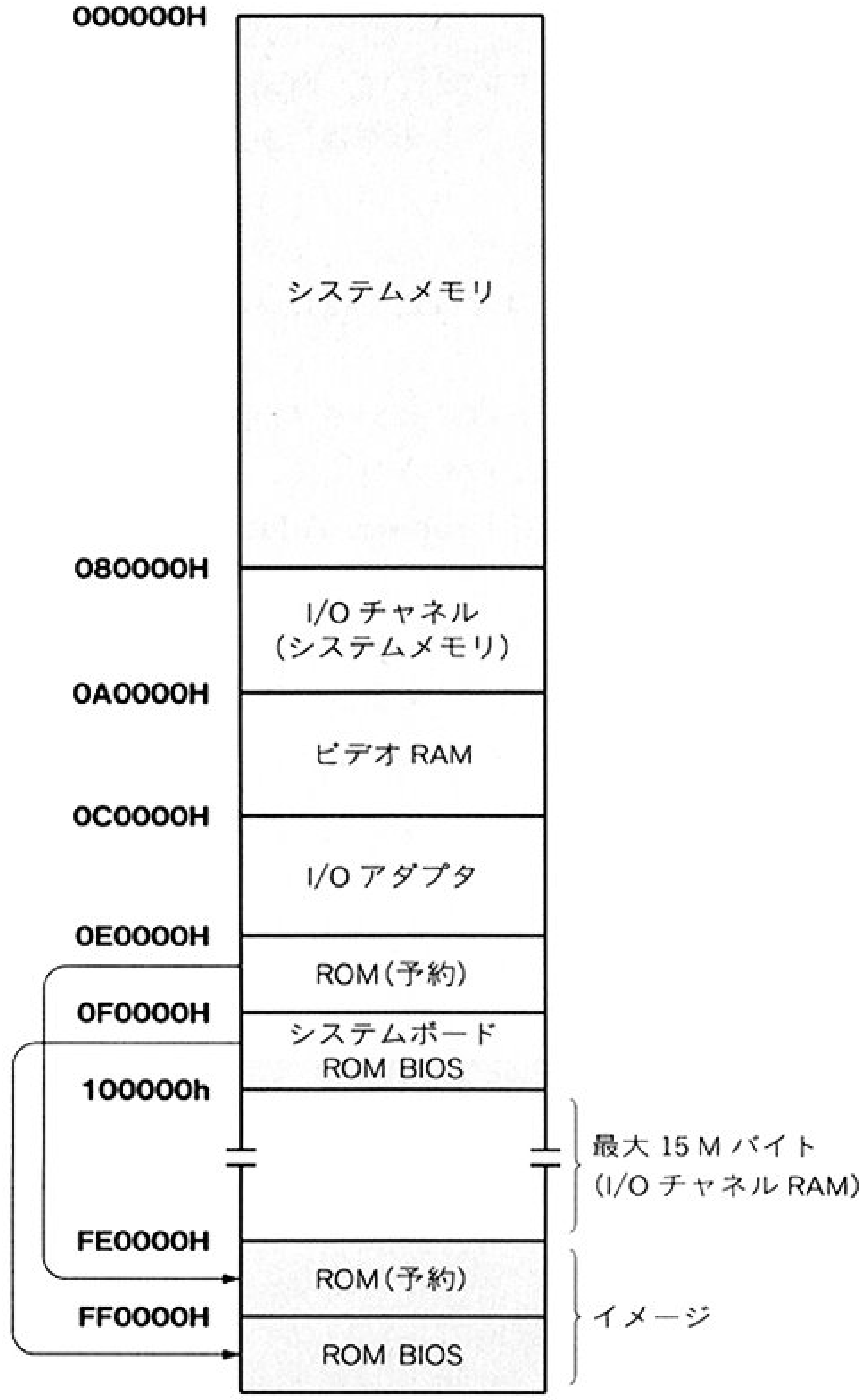


〈写真6〉 XTバス



〈写真 7〉 PC/AT

〈図 2〉 AT のメモリ構成

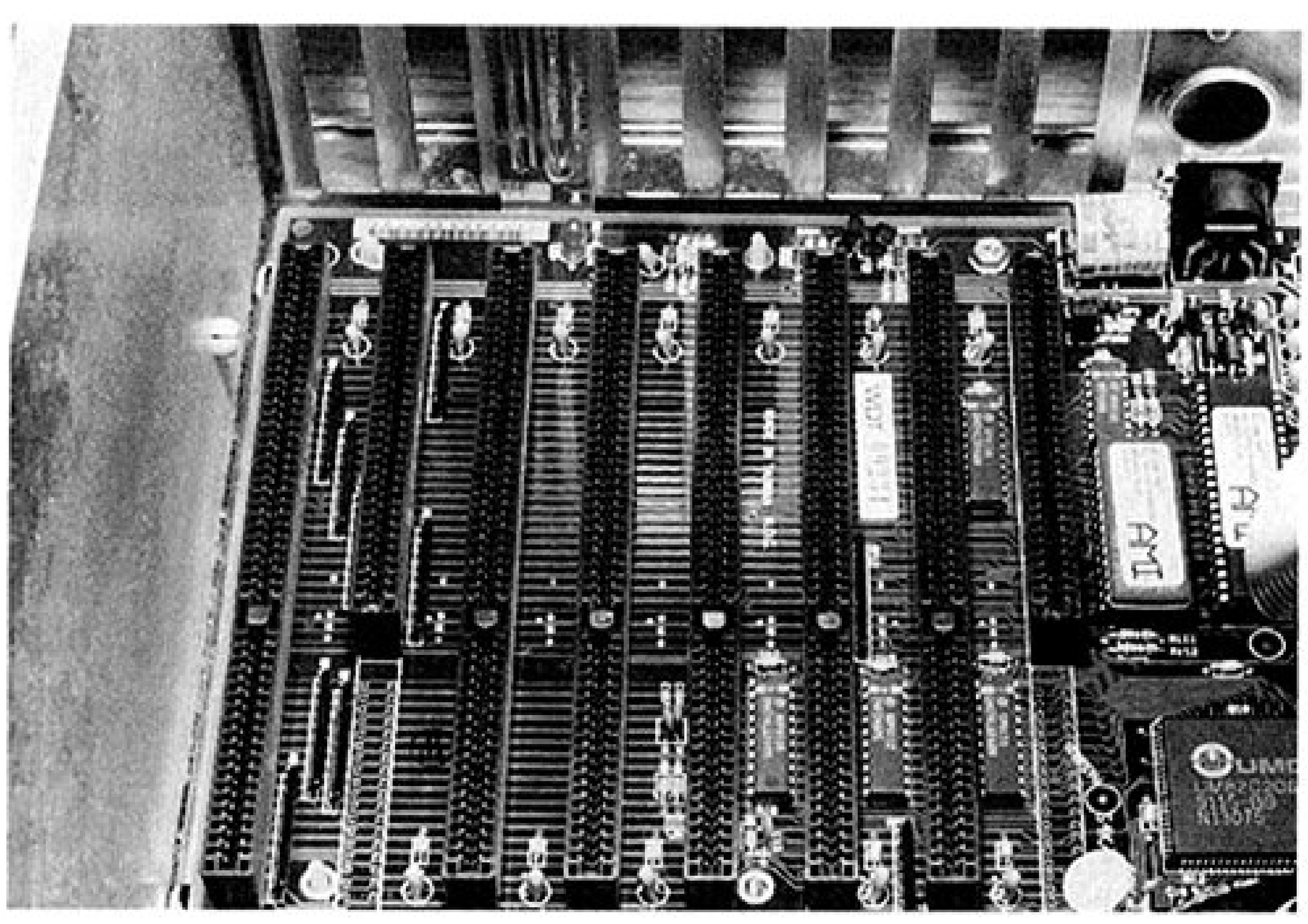


〈表 4〉 PC/AT のスペック

CPU	i80286-6 MHz
タイマー	i8253
DMA コントローラ	i8237A
割り込みコントローラ	i8259A
パラレル I/O	i8255A
標準メモリ	256/512 K バイト
スロット	16 ビット拡張スロット
フロッピディスク	1.2 M バイト
ハードディスク	20/40 M バイト
その他	クロックジェネレータ, BIOS用 ROM, システムの設定情報の記録用 CMOS メモリ付きリアルタイムクロック

隆盛も元をたどればこの PC/AT あってこそだといえます．PC/AT のスペックは**表 4** のとおりです．

PC/AT は，CPU にプロテクトモードとリアルモードという二つのモードをもつ 80286 を使用しましたが，16 M バイトの物理アドレス空間をアクセスできるプロテクトモードでは，従来の 8086/8088 のソフトウェアをそのまま実行することができないため，8088/8086 互換のリアルモードを使用し，高速な 8086 として使用するようになっていました．PC/AT のメモリ構成を**図 2** に示します．



〈写真 8〉 AT バス

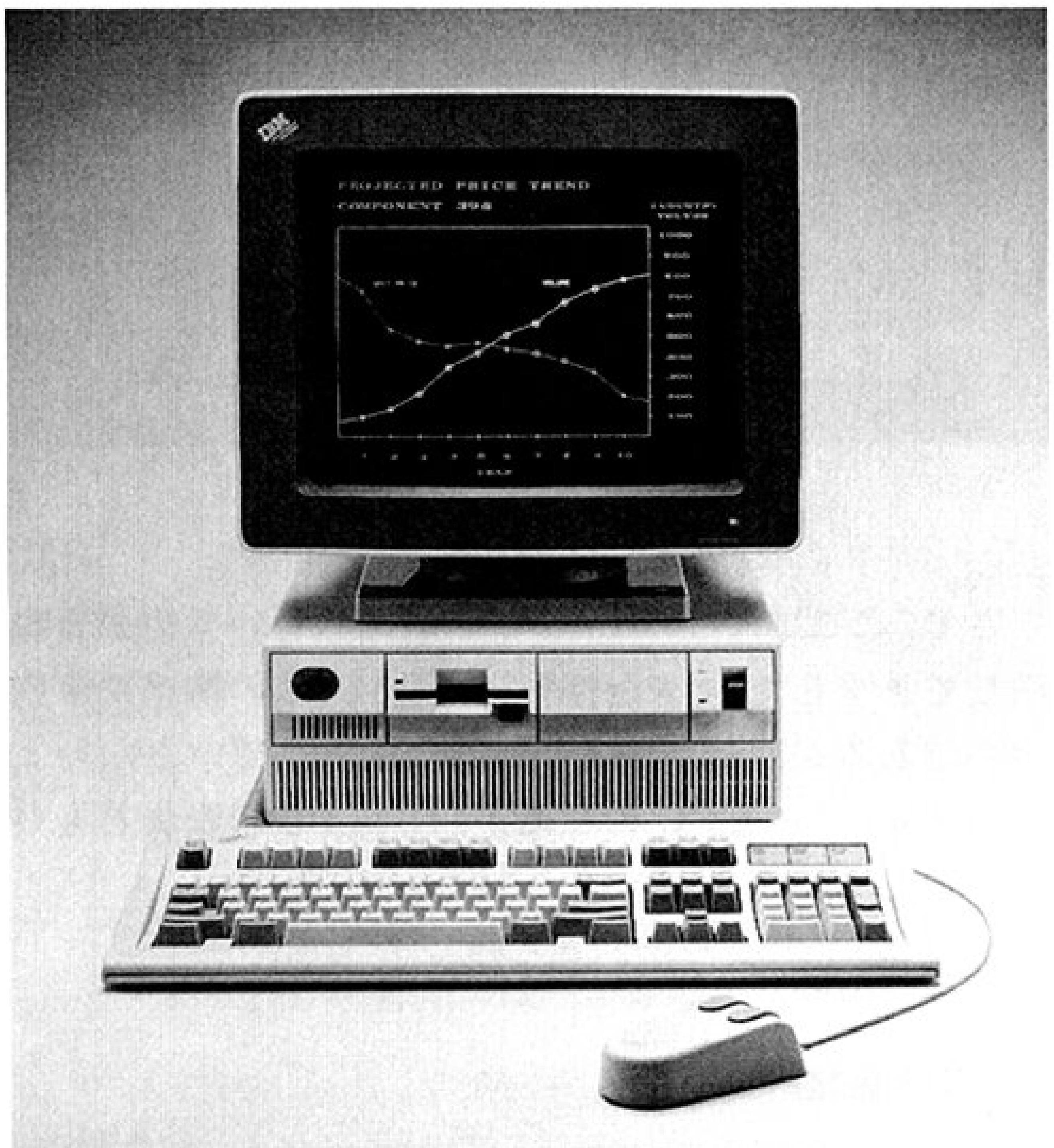
● **バスの拡張**

拡張スロットのデータ/アドレス信号線を拡張するために，PC/AT では従来の XT バスのコネクタに加えて新たなコネクタを増設し，従来の 8 ビット幅のカードもそのまま使用できるようになっています(**写真 8**)．

この XT バスのコネクタと拡張されたコネクタで構成されるバスがいわゆる AT バス，あるいは ISA バスと呼ばれるものです．

バッテリでバックアップされた RTC(Real Time Clock)も PC/AT から採用され，RTC 内部の CMOS メモリの中にシステムの設定情報を記録できるようになりました．

基本的には，現在の PC 互換機はこの PC/AT をもとに性能や機能の向上が図られていますが，この PC/AT を最後に IBM はそれまでの PC の路線から大きな方向転換をすることになります．



〈写真 9〉 PS/2



〈写真10〉 現在の PC(IBM PC 720)

PS/2(写真 9)

● **オープンアーキテクチャからクローズアーキテクチャへ**

IBM は PC/AT まで，オープンアーキテクチャ思想を掲げて，その技術情報をすべて公開していました．

その中には回路図や BIOS のソースリスト，拡張スロットの仕様までが含まれており，それらが多くのサードパーティを産み出して PC 市場を発展させた原動力になりましたが，その反面，多数の PC 互換機メーカーを産み出す結果にもなってしまいました．

実際に 80386 を搭載した PC/AT 互換機を最初に市場に発表したのは Compaq であり，最終的には PC 市場は米国や台湾などの互換機メーカーにほとんど独占されてしまう結果になりました．

そのため IBM は，従来のオープンアーキテクチャ思想に見切りをつけ，ロイヤリティを支払う企業にだけアーキテクチャ情報を公開する PS/2 という新しいアーキテクチャの PC を発表しました．

● **マイクロチャネルアーキテクチャ**

PS/2 では CPU に 80386 を使用し，ISA バスに代わって，新規設計の MCA(Micro Channel Architecture)を備えています．

MCA は，技術的にみればその当時にすでに ISA バスの欠点として指摘されていて，ボトルネックになってきたバス幅や転送速度の改善が図られていたり，マルチ CPU への対応やソフトウェアによるカードのアドレスや割り込みのコンフィグレーションなどをサポートしていました．

しかし，ISA バスとの互換性がまったくなく，サードパーティも MCA の採用に二の足を踏む会社がほとんどであり，さらには ISA バスを採用している PC/AT 互換機がすでに市場をほぼ独占していたため，IBM の力をもってしても主流になることはできませんでした．

● **16 ビット OS/2 の失敗**

また，PS/2 の発表と同時に Microsoft と共同で開発した OS/2 を IBM は強力にプロモートしたのですが，その頃の OS/2 は DOS のアプリケーションとの互換性が非常に低かったのも失敗の原因になったようです．

つまり，技術的に多少優れていても当時の PC 市場はすでに過去のソフトウェアや拡張ボードなどの資産を継承できない機種を受け入れなくなっていたわけです．

現在の PC(写真10)

これ以降の PC の歴史は，PC 互換機メーカーによる競争原理が働いたおかげで PC/AT 互換機が市場にあふれかえり，Intel から供給される新しい CPU が次々と採用されていきました．

チップセットの普及やメモリの大容量低価格化によって，現在では Pentium を搭載したワークステーションクラスの PC まで登場するにいたってます．

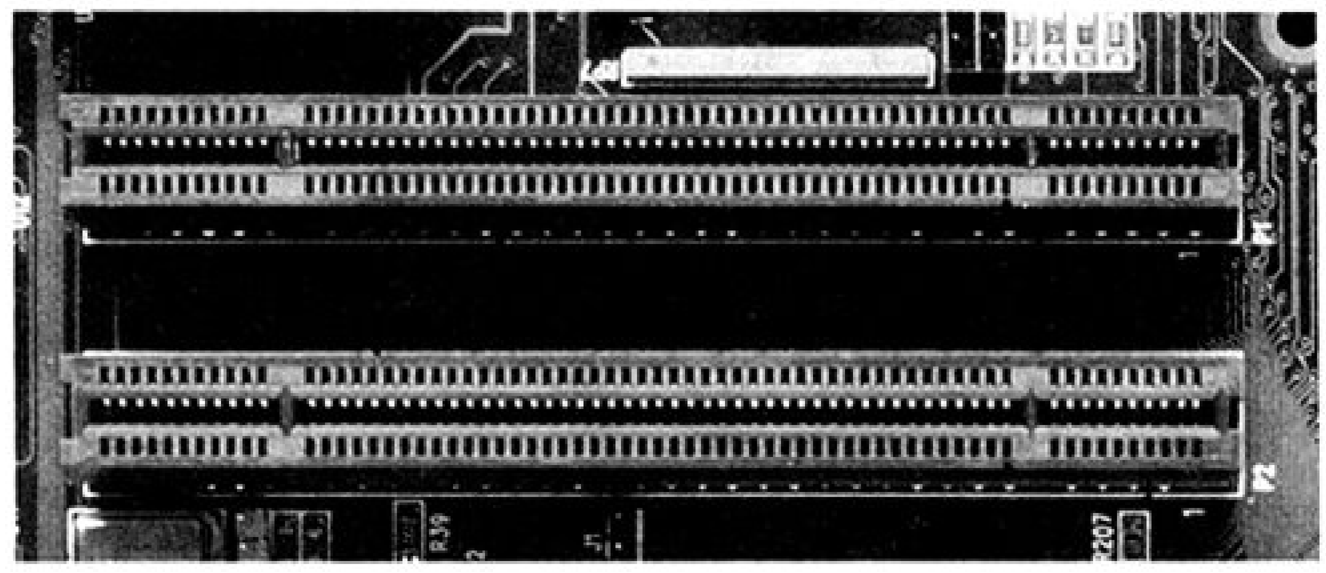
高機能なチップセットが低価格で普及しているため，現在の PC のマザーボードには CPU の他には数個のチップセット LSI と若干の周辺回路の IC が搭載されているだけです．

また，ISAバスの性能がCPUに追い付かなくなっているために，ISAバスを拡張して転送速度の向上を図り，MCAに対する互換機メーカーの対抗手段ともなった，EISAバスが登場して主流になるかと思われました．

しかし，現在ではVLバスと呼ばれるCPU直結タイプのローカルバスとPCIと呼ばれる新しい設計のバスが主流になっています(**写真11**)．

PCIバスはISAやEISAと互換性はありませんが，PCI採用のマザーボードはすべてPCIとISAの両方を持っており，転送速度やバス幅の要求が低いものや今まで使用していたカードがそのまま使用できます．

また，中にはPCI, ISA, VLの3種類のバスを持っているボードもあります．



〈写真11〉 PCIバス

さて，ここまではIBM PCの歴史について概要を説明してきましたが，これからはIBM PCの構成要素の中心であるCPUについて個別に説明していきたいと思います．このCPUの変遷が，IBM PCの歴史ともいえるでしょう．

マイクロプロセッサの進化

8086以前(1970年代初頭〜1978年) 8ビットPC全盛期

● Z80の登場

8086を発表する前にも，Intelは最初のマイクロプロセッサとして4004を発売して以来，8008や8080，8085といったプロセッサを開発してきました．

しかし，この80系CPU市場の成長を決定的にした牽引車は，何といってもZ80でしょう．

Z80はZilogが開発した8080互換のCPUで，8080の命令セット，割り込み機能およびレジスタセットなどが拡張されていました．

周辺回路も少なくてすみ，Z80ファミリの周辺LSIも早い時期から供給されていたため，一気にその当時の8ビットマイコン市場のベストセラーの一つとなりました．今でもコントローラやワンチップマイコンのコアとして使用されています．

● AppleIIに採用された6502

一方，MotorolaがMC6800(**写真12**，68000ではない)を開発し，6805や6809などの68系MPUを開発しましたが，80系に比べると市場規模は非常に小さかったようです．

例外的な68系MPUは，6800の動作を高速化した6502(MotorolaではなくMOS Technologyが発表)で，これはAppleIIなどに使用されました．国内でも任天堂がゲームマシンに使用しました．

● CP/Mマシンの隆盛

この頃の基本ソフトはBASICが多く，オプションとしてCP/M，あるいはOS-9などのOS(ほとんどがフロッピベース)を使用していました．

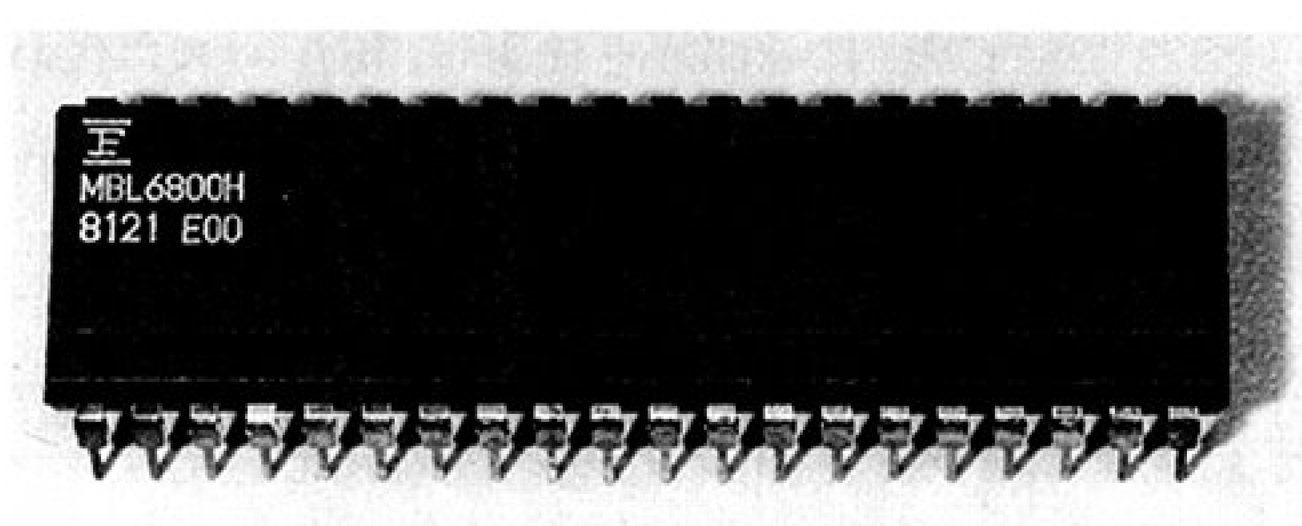
8086が登場した後の数年間も，パソコンに使用されるCPUの大半はZ80であり，Motorolaの68系CPUは6805や6811などのコントローラ向けCPUとして発展していきます(筆者は学生時代にZ80のパソコンとCP/Mでゲームソフトを作っていました．徹夜の連続で，フロッピのシーク音が目覚し代わりでした．懐かしいなぁ)．

また，この頃のパソコンのアーキテクチャは各社各様であり，Z80搭載でCP/Mが動作するコンピュータであってもメモリ構成やI/Oの種類なども各社各様で，アーキテクチャの違いがセールストークの一つになっていた時代です．

なお，ほとんどの8ビットCPUは内部のレジスタ構成としてアキュムレータ，インデックスレジスタ，プログラムカウンタ，スタックポインタ，フラグレジスタなどを持っていましたが，8ビットのアキュムレータだけに計算結果を入れることができ，アドレッシングモードも非常に単純でした．

プログラムカウンタやスタックポインタなどのメモリを指し示すレジスタは16ビットで，アドレス空間は$2^{16}=65536=64$ Kバイトが一般的でした．

CPUのビット数をいう場合，この当時はアキュムレータのビット幅かデータバス幅を指して言うことが多



〈写真12〉 6800

く，8080は純粋な8ビットCPUでしたし，6809は16ビット(8ビット二つに分割できる)のアキュムレータを持っていたため，8/16ビットMPUなどと呼ばれていました．ちなみに，技術的に言うとALUのビット幅がそのCPUのビット数となります．

8ビットから16ビットへの移行

● 過去を継承すべきか，見切るべきか

さてこの後に，16ビットCPUへの市場の要求と半導体製造技術の進歩により，Intelをはじめとする CPUメーカーが従来の8ビットCPUから16ビット CPUに移行してくるわけですが，ここで16ビット CPUを開発する際に後々まで影響してくる2種類の選択肢がありました．

一つはIntelが選んだ従来の8080の8ビットCPUのアーキテクチャを拡張する形で16ビットにする方法と，MotorolaやZilogなどが選んだ新規設計の16ビットCPUを開発する方法です．

● Intelは過去を継承した

Intelが新規アーキテクチャを採用せずに8080を拡張して16ビットCPUを作成した理由は，その頃は前述のように80系CPUの市場が(Z80のおかげで)大きく成長しており，そこで使われてきたソフトとハードの資産をなるべく継承する形で16ビットに移行(市場を奪回)する必要があったからです．

● Motorolaは過去を見切った

Motorolaの場合はPC市場での8ビットMPUに見切りをつけ(8ビットPCでは全滅に近い状態だったため失うものが小さく，8ビットMPUはコントローラ向けに特化していった)，68000(**写真13**)のように32ビットCPUへの移行をも見据えた新規設計の16ビットMPUを開発したわけです．

そして，Motorolaの戦略は68020が出たあと，ワークステーション市場で花開くことになりますが，その後のRISCの台頭によりワークステーション市場から淘汰されることになり，68000系は今ではZ80的な使われ方(コントローラやコア)に活路を見出すことになります．

8086の登場と8088

● アドレス空間の確保〜セグメンテーション〜

さて，Intelは1978年に16ビットCPUとして8086を登場させたわけですが，8086ではレジスタの幅を16ビットにする方法としてレジスタペアを採用し，二つの8ビットレジスタか，一つの16ビットレジスタとして使用できるようにしています．

また，アドレス空間を拡張するためにセグメンテーションという技法を用いています．これは二つのレジスタのうち一方の値を4ビットシフトして加算した結果をアドレスとして使用する方法です．

図3に8080と8086のレジスタ構成を示します．これが，i386の登場まで世間の非難を浴びていた「64Kバイトの壁」の原因でもあるわけですが，8086が登場した時代背景からすれば，この方法を採用するのがもっとも市場に受け入れやすかったのは事実でしょう．

8086の最大の目的は，8080でスタートしてZ80に占領されてしまった8ビットマイコンの市場を奪い返すことでしたから，この方法が最適だったのだと思います．

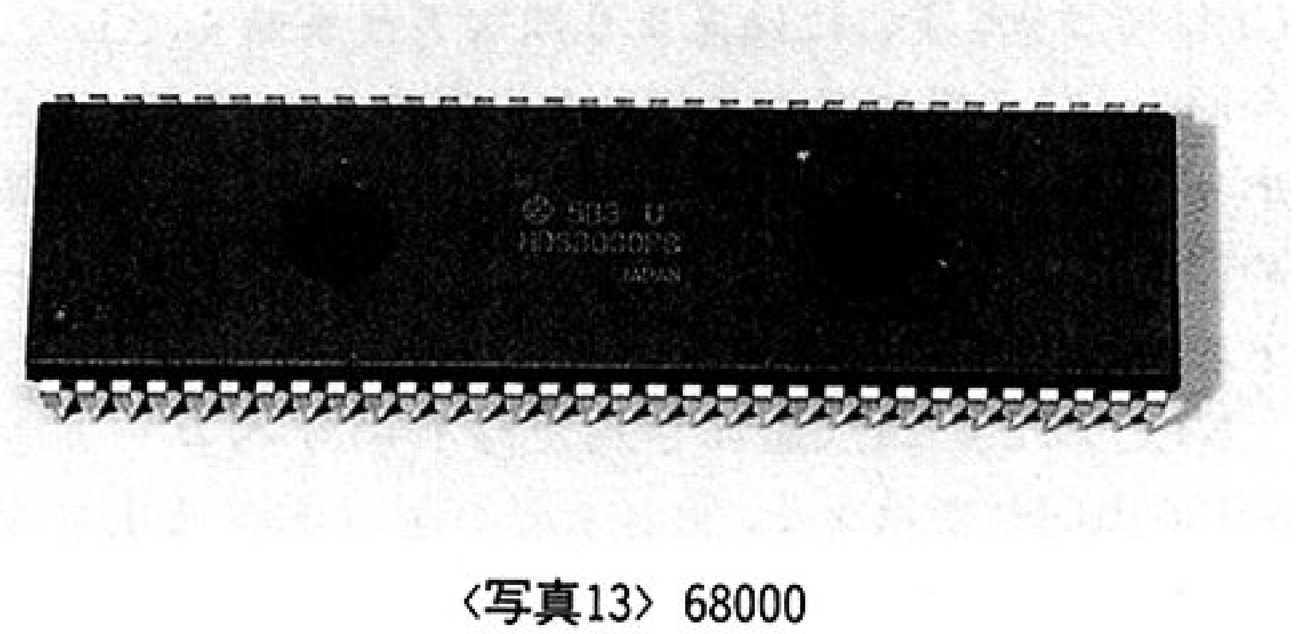
● 8087の登場

また，このときにすでに浮動小数点コプロセッサとして8087が登場していたことも特筆すべきでしょう．

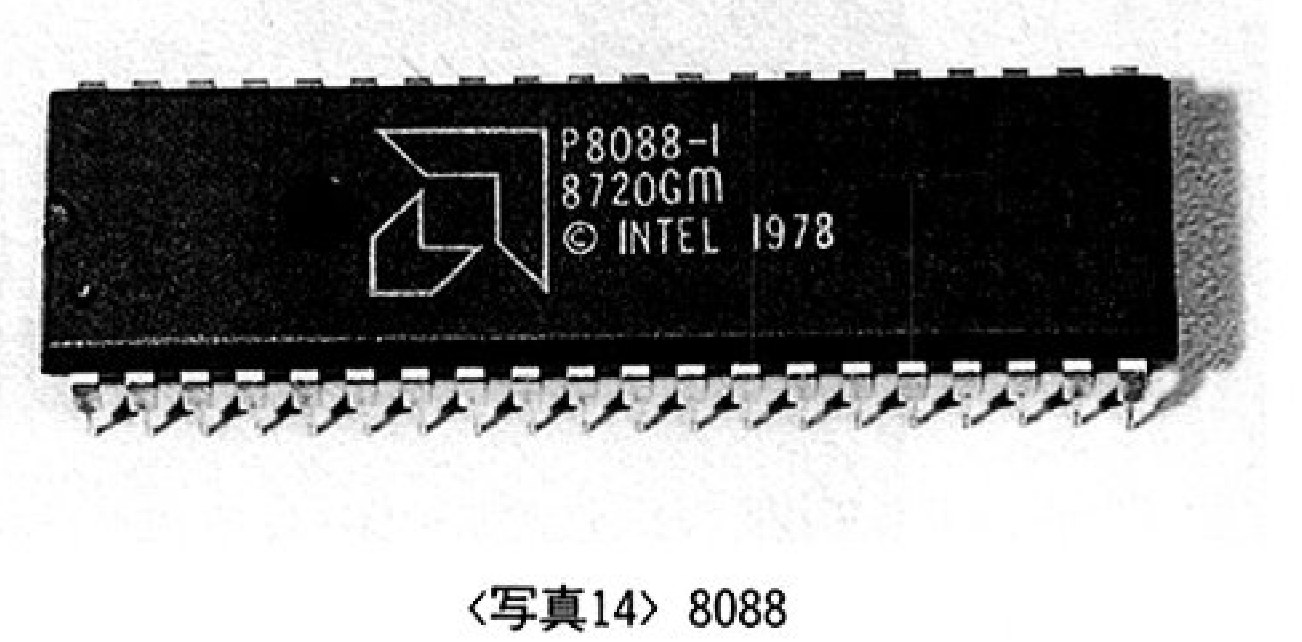
この8087が存在したおかげで，PCがビジネスやホビーだけではなく，小規模な科学技術計算にも使用できるようになったわけで，浮動小数点演算を大量に行うアプリケーションでは必須のアイテムとなり，ユーザーのステータスシンボルともなりました(おれのPCにはコプロが入っているから速いんだよ，という言い方ができた)．

● 8088の登場

少し遅れる形で8088というCPUも登場しています(**写真14**)．当時はまだメモリの容量が小さく高価であったため，データバス幅が16ビットの8086のアドレス空間である1Mバイト全部をメモリで満たすことは経済的にも市場の要求からも(そんなに大きなアプリケーションもありませんでしたから)無理がありました．

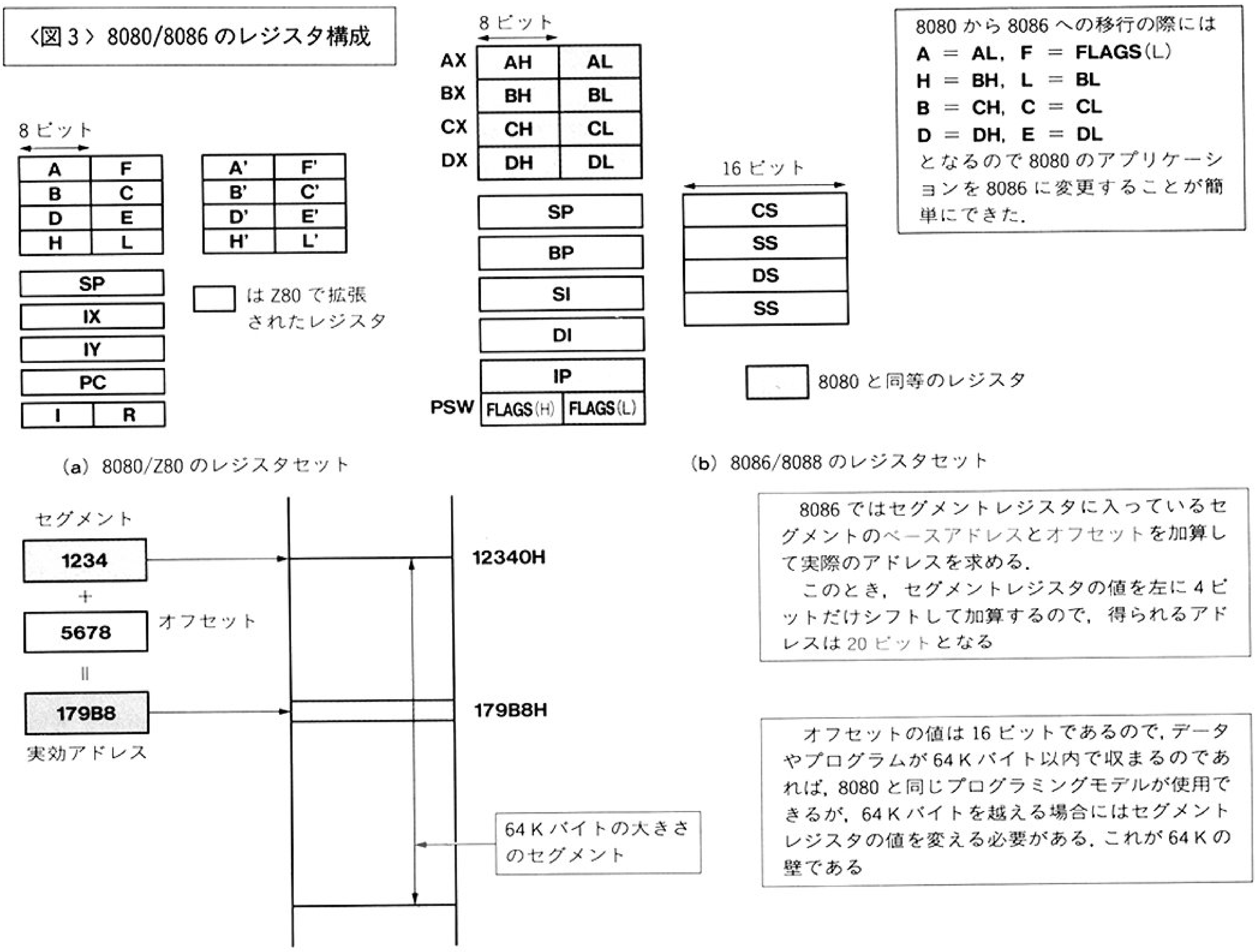


〈写真13〉68000



〈写真14〉8088

〈図3〉8080/8086のレジスタ構成

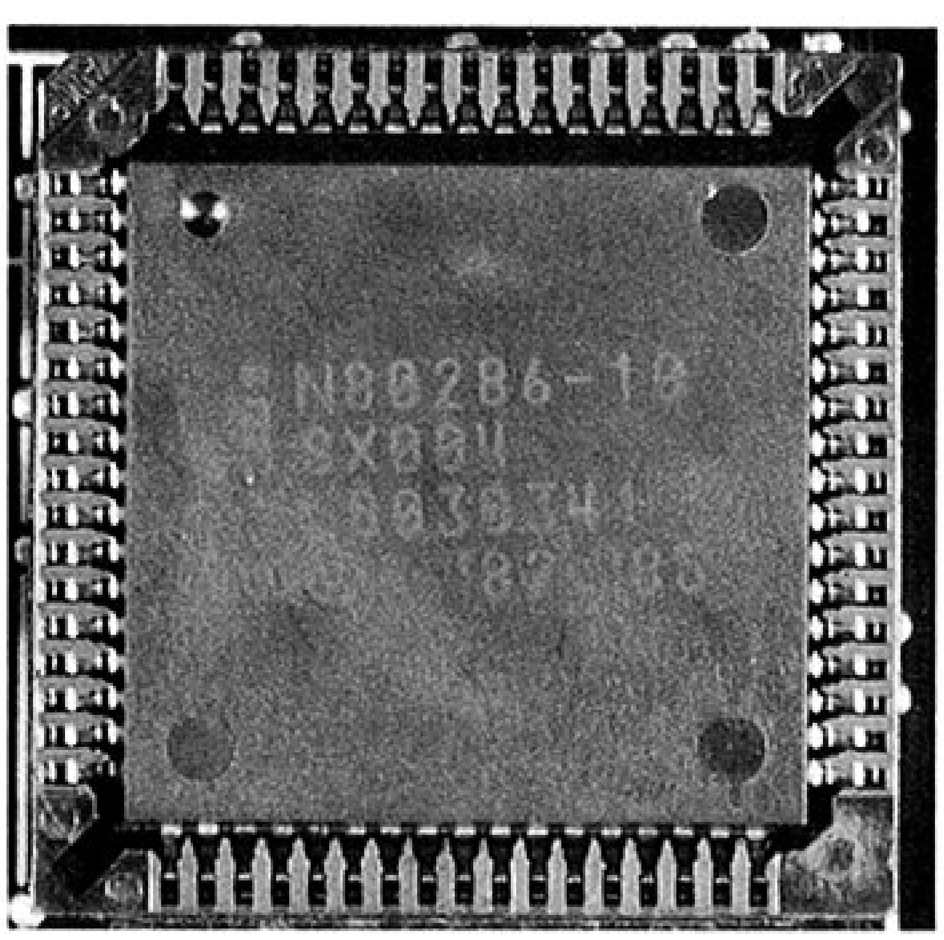


(c) 8086でのセグメンテーションのしくみ

また, 8086は16ビット幅のメモリ構成(EPROMは8ビット, DRAMは1ビット幅でした)を必要としました. そこで, データバス幅だけを8ビットにして接続するメモリの部品数を少なくできる8088も登場したわけです.

ただし, 8088は本来の8086が16ビットデータを1バスサイクルでアクセスするのに対し, 2バスサイクルでアクセスするわけですから性能的には8086よりも若干劣ってしまいます.

〈写真15〉80286



ここで若干というのは, 8086の命令長が1バイト, つまり8ビットのものが多く, CPU内部の処理速度は同じですから, きっかり半分に性能が落ちるわけではないということです.

実際, 8086の6割ぐらいの性能を出すことが可能で, ボードの部品点数も大幅に減らすことができるわけで, コストパフォーマンスと競合メーカーの8ビットマイコンとの差別化においては, IBMがこの8088を最初のPCに採用したのも理にかなった選択だったのでしょう.

● 80186とV30

80186は8086を使用する場合に必ず必要となる周辺回路を内蔵し, 若干の命令追加と性能向上を図ったCPUですが, PCにはあまり採用されませんでした. 今回のPCの歴史でもその名前を残すことはまったくありませんが, コントローラとして今もその名前を留めています. 例えば, SCSIボードやイーサネットカードに使われています.

NECのV30は, 追加された命令のオペコードが80186と同じではなく, 実行速度の速い8086として主にPC9801のために使われたCPUでした.

80286

● 8086 のセグメントを継承

さて，8086 によって 80 系 CPU の市場の奪回に成功した Intel は，この後次々と 8086 の機能拡張に取り組みます．そして，1982 年にアドレス空間の拡張とメモリ保護機能，それにマルチタスクへの対応を追加した 80286 を登場させました(**写真15**)．

それまでの 8086 でのアドレス空間は 1 M バイトしかありませんでしたし，メモリ管理機能もありませんので，Unix のような OS を動作させるにはそれらの機能が必要だったわけです．初期の OS/2 もこの 80286 の機能を前提に開発されました．

アドレス空間を拡張するには単純にレジスタを 32 ビット幅に拡張する方法もありましたが，Intel はセグメンテーションのメカニズムを変更することによって物理メモリ空間を 16 M バイトに拡張することにしました．

セグメントの指定方法を拡張してメモリ空間を拡張したために，依然として 1 セグメントの大きさは 64 K バイトのままでした(**図 4**)．

● MS-DOS が動作しないプロテクトモード

当時でも 32 ビット化への要求がありましたが，Intel はメモリ管理機能を 8086 アーキテクチャに追加することを優先したわけです(この点は 68020 との対比をすると興味深いものがある)．

ただし，80286 の機能をフルに活用するにはプロテクトモードという新設された動作モードで動作させる必要がありました．

このモードでは，今までのセグメントレジスタがセグメントセレクタとして使用されます(8086 ではセグメントレジスタにはセグメントのベースアドレスの上位 8 ビットを入れていたが，プロテクトモードの 80286 ではセグメントセレクタにセグメントの番号と属性を入れるようになる)．

プロテクトモードでは，当時主流となっていた MS-DOS がそのまま動作しませんので，80286 は従来の 80186 との互換性をもち，高速な 8086 として動作するリアルモードを備えていました．

実際の PC 市場では，80286 はほとんどリアルモードで MS-DOS を動作させ，メモリ管理用ドライバがプロテクトモードにスイッチして 1 M バイト以上のメモリを管理する方法が一般的でした．

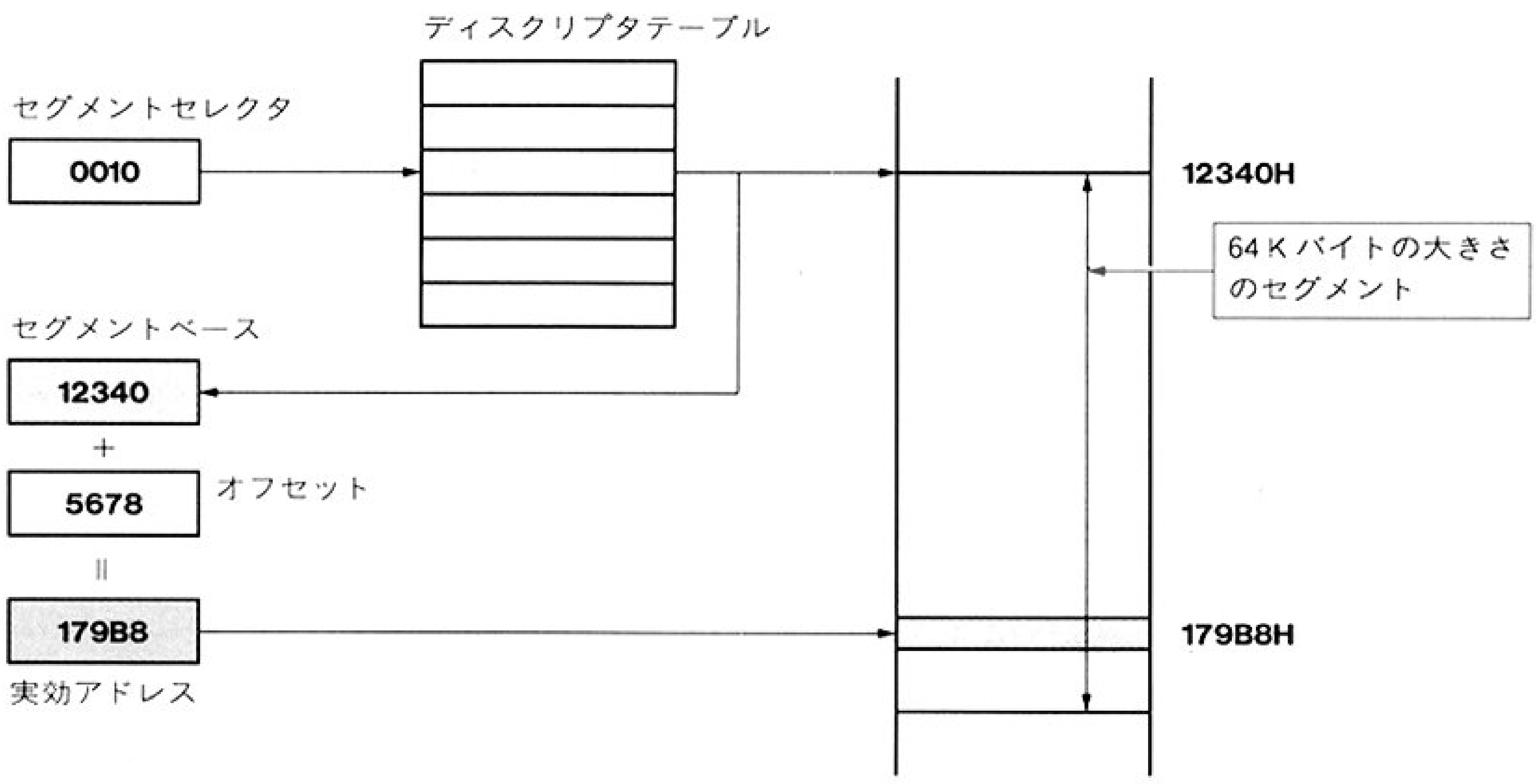
ただし，80286 ではリアルモード→プロテクトモードへの切り替えはソフトウェアで可能なのですが，プロテクトモード→リアルモードへの復帰はハードリセット以外に方法がなく「ギヤチェンジのたびにエンジンを止めなくてはならない」ために，リセット前の CPU の状態を保存してからリセットをかける回路が PC のマザーボードに設けられていました．

● HMA の存在

また，8086 アーキテクチャではセグメントアドレスとオフセットを加算して実際のメモリアドレスを求めるわけですが，その時の加算で桁あふれが発生した場合に 8086 では 1 M バイト内のメモリをアクセスしますが(アドレスが A19 までしかないので)，80286 以降では 1 M バイト以降のメモリをアクセスしてしまいます(A20 が“1”になる)．

この機能は今では HMA として一般化しています

〈図 4 〉
80286 のプロテクトモードでのアドレス生成



80286 ではセグメントレジスタに入っているセレクタ値がディスクリプタテーブルへのインデックスとなり，選択されたディスクリプタに入っているセグメントアドレスとオフセットを加算して実際のアドレスを求める．
ただし，オフセットの値は 16 ビットであるので，64 K の壁は 80286 では依然として残ったままである

が，PCのマザーボードには機能を殺すためにアドレス信号線のA20を強制的に0にする回路が設けられていました．

なお，80286とOS/2により，市場は一気にマルチタスクOSに向かうと思われましたが，ふたを開けてみるとIBMの「心変わり」により，市場は依然としてMS-DOSのまま成長し，MicrosoftはMS-DOSの機能拡張とMS-Windowsの開発に着手することになります．

● PC市場以外では

ただし，PC市場とは別に成長してきたワークステーション市場や組み込み市場ではセグメンテーションを使用せずにフラットな4Gバイトアドレス空間がアクセスできる68020がシェアを独占していました(Sun3がその代表でした)．

仮に，この当時にIntelが32ビットの80x86を出していれば今頃は世の中はIntel一色だったかもしれません．

〈写真16〉
80386



コラム●80286のほうが80386/386SXより速い？

その昔，といっても6年ほど前ですが，AMD社が80286，80386，80386SXをクロック16MHzで評価したレポートがありました．

そのレポートでは，80286のほうが16ビットの命令を速く実行できるということで，80386と比較を行い，

▶ 80286のほうが速く実行できる命令の数…74
▶ 80386のほうが速く実行できる命令の数…50
▶ どちらも同じ命令の数……………………66

という結果を出しています．

その結果を理由に，「80386の16ビットバス版80386SXは，80286に比べなんらメリットはない」とAMD社は結論付けていました．

その AMD社も，今ではIntel互換の32ビットCPUチップの供給をしているメーカのひとつです．

80386(i386)

● 86アーキテクチャの完成

80386の登場により，80x86アーキテクチャは一つの完成を見たといってもよいでしょう(**写真16**)．

レジスタは32ビット幅に拡張され，一つのセグメントで4Gバイトのアドレス空間をアクセスできるようになりましたし，80286と同様のプロテクトモードでメモリ管理を行うこともできます．

また，コード/データ/スタックのセグメントをすべて同一のものにすれば，68020などのほかの32ビットCPUと同様の使い方(フラットモデルと呼んでいる)ができるようになったこともPC以外(組み込みやUnix)の市場では大きなメリットです(**図5**)．

また，MMUにデマンドページング機能が付いたことも画期的なことで，これによりスワッピングを行うマルチタスクOS(Unixなど)への対応がより一層効率よくなりました．

MS-DOSでは80386のメモリ管理機能を使ってUMBやEMSを仮想的に作り出すことが可能になりました(**図6**．ただし，今ではこの方法が一般化している)．

また，新設されたバーチャル86(V86)モードにより，複数のDOSアプリケーションごとに論理的に1Mバイトのアドレス空間を割り付けて動作させることも可能になりました．

● PC市場の方向性が決まった！

この80386とWindows3.1により，PC市場におけるIntel＋Microsoftの地位は盤石のものとなったようです(市場の地位に盤石なんて本当はありませんが，他社を寄せ付けるスキがなくなったことは事実です)．

ただし，実際に使用されていたアプリケーションはMS-DOSの16ビット用のものがほとんどで，80386では32ビット用に同じプログラムを作成するとプログラムサイズが大きくなるばかりでなく，実行速度も遅くなることをほかのコンパチCPUベンダー(AMDがAm80286のセールストークとして盛んに言っていた．コラム参照)が主張していました．

このときに，AMDがAm80286の33MHz版などを登場させていれば，市場はもっとエキサイティング(混沌？)だったでしょう(ほとんどのCPUベンダーは高速化への努力を最新CPUのプロセスにしか応用しないんですネ)．

● 蛇足ですが

今でも，50MHzのZ80や100MHzの68020なんかがあればCPU市場はほとんどそれらでことが足りるはずです(しかし，そのCPUは顧客に買い叩かれるので，そんなことをしても儲からないんですよネ．最

〈図5〉
80386の主要なレジスタマップ

汎用レジスタ

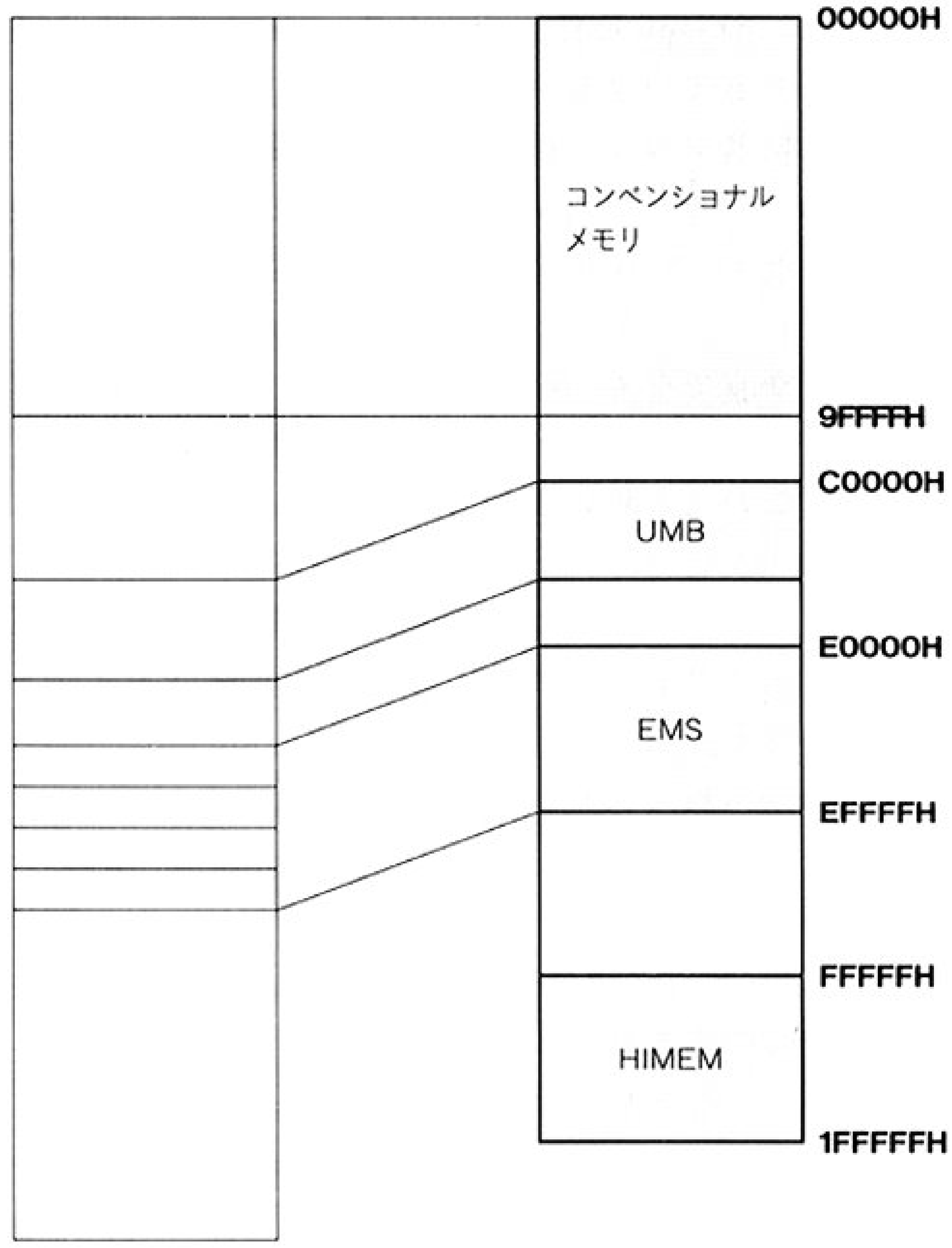
31–16	15–8	7–0	
	AH	AL	EAX
	BH	BL	EBX
	CH	CL	ECX
	DH	DL	EDX
	SI		ESI
	DI		EDI
	BP		EBP
	SP		ESP

(AX, BX, CX, DX: 15–0)

インストラクションポインタ，フラグレジスタ

31–16	15–0	
	IP	EIP
	FLAGS	EFLAGS

〈図6〉ページ式MMUによるUMBとEMSの実装



i386以降のCPUでは，ページ式MMUにより4Kバイトのページ単位でリニアアドレスを物理アドレスに変換することが可能になり，UMBの割り当てや，従来ではEMS用拡張ボードなどで実現していたEMSの実装などはMMUのディスクリプタテーブルの設定を変更するだけで可能になった．

MS-DOSの**HIMEM.SYS**や**EMM386.EXE**などのメモリ管理用ドライバがこれらの機能を実現している．例えば，MS-DOSアプリケーションがUMBエリアの**C0000H**番地をアクセスしても，実際にはそのアドレスがMMUによって変換され，実際にCPUの領域に割り当てられている番地のメモリをアクセスする

新CPUの方が殿様商売をしやすいですからネ）．

なお，80386はリアル/プロテクト/V86の三つのモードを持ちますが，それらのモード間の移行はすべてソフトウェアで使用可能となりましたし，A20の機能も80286では必要だった外部回路がCPU内部に設けられています．

読者の方の中には80386で完成したセグメント＋MMU＋デマンドページングに嫌悪感をおぼえる人も少なくないと思いますが，従来のバイナリ互換性を維持した上で新しい機能を追加することに成功したIntelの技量には賞賛すべきものがあると思います．

また，この頃からCPUの名前がi386という言い方になったような気がします．

i486（80486）

● 386の高速版

i486は80386でほぼ完成した80x86の基本アーキテクチャをそのまま継承しながら，より一層の高速動作と集積度を追求したCPUです(**写真17**)．

1チップに集積できるトランジスタ数が加速的に増えたおかげで，内部キャッシュやFPUを内蔵したり，パイプライン動作を取り入れることによって高速処理を図っています．

i486が出てからはCPUクロックがどんどん高速になり，16ビットコードのほうが速い(同一クロックなら今でもそういう場合もある)から16ビットCPUのほうがよいなどと言う競合他社の言い分は通りにくくなり，色々な競合メーカーがi386コンパチやi486コンパチのCPUの開発をするようになりました．

● オーバードライブの登場

今のところミッドレンジ，あるいはポータブルPCにはほとんどi486がスタンダードとして使用されて

〈写真17〉
i486DX2



います．

Intelは，ユーザーの用途に合わせていろいろなバリエーションのi486を作っています(筆者もすべての種類を把握できていません)が，オーバードライブという発想と，倍速クロック技術が大きな特徴でしょう．

オーバードライブという形で既存のCPUを取り替えさせて性能アップを図るというやり方は，それまでのアップグレード＝買い替えという図式からユーザーを救ったばかりでなく，Intelにとっても一度顧客になったユーザーを他社に持っていかせない戦略として非常に的を得ていたと思います．

また倍速技術は，マザーボードの動作クロックスピードがそのままでも最高で2倍の性能アップになるのですから，これもユーザーにとっては大きなメリットがありました．

この頃から秋葉原やシリコンバレーのパソコンショップではCPUを直接ユーザーに販売することが多くなりましたし，マザーボードもCPUの抜き差しが容易なZIF(Zero Insertion Force)ソケットを搭載するのが標準になりました．

"Intel Inside"というコマーシャルも，ユーザーにオーバードライブや倍速CPUを購入させる戦略として出てきました(ちなみに，筆者の手作りPCはその戦略にまんまとはまり，486DX-33，DX2-66，そしてDX4-100と進化している．これで，本誌の原稿執筆やゲームには十分すぎる性能なのだが…．やはりCPUパワーは麻薬である)．

また，i486を使用したPCのマザーボードから外部キャッシュメモリを搭載することが常識になりました．

また，i486のバスをそのまま使用する形で高速バス動作を可能にしたVLバスが出てきて，VLバスに対応したWindowsアクセラレータ機能付きスーパーVGA(SVGA)ボードでWindowsの画面描画が画期的に高速になりました．

Pentium

● Pentiumの語源

さて，i486の頃から競合メーカーもいろいろなi486互換チップを登場させてきて，極端な場合には下手をすると80586という製品番号を他社が先に使ってしまう可能性が出てきました．

またIntelは，xxx486という名称の互換チップの存在を苦々しく思っていたようです．そのために5を示すPentaと，原子の名称に使われるiumを合わせてPentiumという造語を作り，これをi486の後継CPUの製品名としました．

● CISCとRISCの融合

Pentium CPUは，68020からワークステーションやグラフィックアプリケーション市場を根こそぎ奪い取ったSPARCやMIPSに代表されるRISC技術，複数の命令を一度に実行できるスーパースカラ技術を取り入れ，マルチプロセッサにも対応するCPUです．

Pentium CPUについては，現在の主流CPUであることから読者の方々に改めて詳しく述べる必要もありませんが，80x86や680x0に代表されるCISCの性能向上がピークになったと主張するRISC陣営に対して，80x86の性能向上が依然として達成されている(**図7**)ことを証明したCPUといえるでしょう〔ちなみに筆者は，Intel(米国)の隣のビルの会社に勤めているが，この業界を代表する頭脳達がカラッとしたカリフォルニア気質のもとで，敷地内にあるアスレチックジムやビーチバレー場でリフレッシュしながら仕事をしているのを見ると，80x86は殺しても死なないと本当に思う〕．

そしてP6，P7

● P6の技術

この原稿を書いている段階ではP6の正式名称が何になるかわかりませんが，Pentiumよりもさらに進んだスーパースカラ技術，レジスタの動的割り付け，同一パッケージ内で1次キャッシュ内蔵のCPUのダイと256Kバイトの2次キャッシュのダイをCPUのダイのバッグエンドバスという専用の信号線にボンディングしていることなどが非常に目立つ点です．

これにより，PCマザーボートにキャッシュメモリを搭載しなくても，かなりの高性能を出すことが可能で，P6採用のPCのコストアップを最小限に抑えることができるでしょう．

また，パイプライン段数も多くなり，従来のx86の命令をRISC風の命令に分解し，Out of order実行ユニットと動的に割り当て可能なレジスタを使用した高速処理を可能としています．

ブランチによるパイプラインの破棄の損失を最低限に抑えるため，分岐予測キャッシュも強化され，条件moveなどの新命令も追加されています(C言語の戻り値の代入などに効果があるようです)．

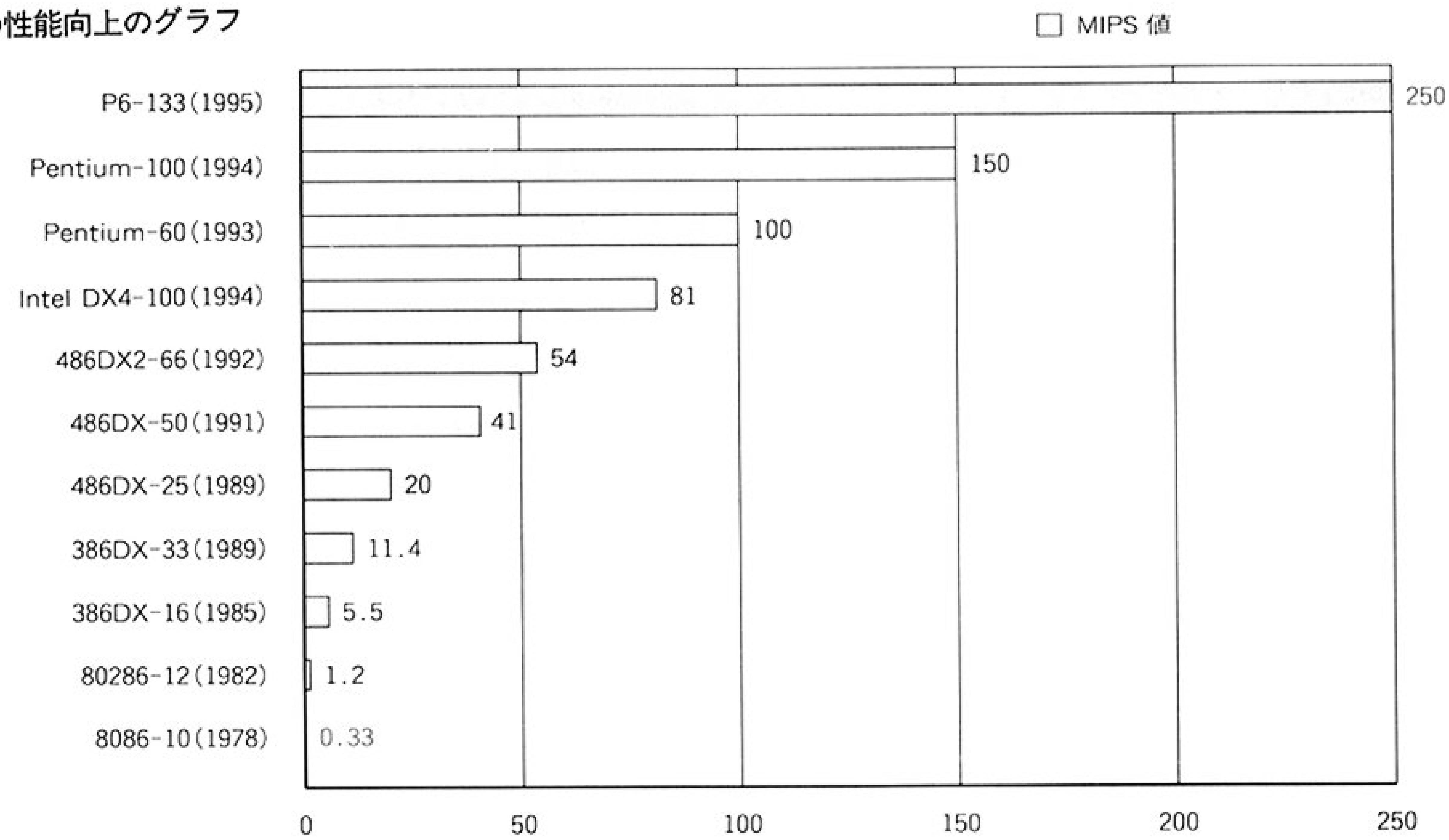
P6に関しては，PCの歴史というより未来に属する範疇ですので，詳しい紹介は別の機会に譲ります．

8086からスタートしたアーキテクチャを維持しながらP6のようなCPUが生まれて来るまでCPUの歴史を顧みて，「あの時IBMが8088を採用していなかったら」とか，「もしも初代PCが68000だったら」と考えると，今のPC市場はまったく違ったものになっていたでしょう．

● P7は，HPとの共同開発

P7はHewlett PackardとIntelの共同開発による，

〈図7〉80x86の性能向上のグラフ



x86のコードとPA-RISCとの両方の互換性をもつVLIW(Very Large Instruction Word)アーキテクチャのCPUだそうです．

VLIWでどうやってx86のコードが走るのか現時点で推測の域を出ませんが，シリコンバレーを代表する両社の共同開発の行方に注目が集まっています〔実際に両社は，フリーウェイ101で15分ぐらい(100 km/hで)の距離にある．それに対して，三社とも別の州にバラバラにいるPowerPC陣営がどう出るかが興味深い〕．

互換チップ

● 互換チップが本家を凌駕することは不可能か

x86の互換チップには，AMDを初めCyrix，NexGen，そしてIBMなど数社あり，なかにはP6で採用されているOut of order実行ユニットやレジスタの動的割り付けなどを採用してPentiumと同等の性能を出しているものもあります．

筆者の個人的な意見ですが，今となっては「Intelよりも速く動作するCPUを，Intelよりも早く市場に出す」ことは不可能でしょう．

その理由は，もしIntelよりも高速に動作するCPUを他社が登場させ，それにIntelが値下げで反撃してきたときに，膨大な開発費を補うだけの利益を競合メーカーが得られるかという問題(実際のところ，CPUの価格はほとんどが開発＋裁判費用？)があるからです．

また，どの会社も優秀な頭脳を集めて同じことをやっているのですから，先行しているメーカーが有利なことに変わりはないという覚めた見方もできます．

ただし，技術的にはIntelを陵駕するCPUを競合メーカーが開発する(あるいはIntelが大失敗を犯す)ことは可能なのであり，性能や価格がIntelのCPUよりも圧倒的に優れていれば市場を脅かすこともできるでしょう．

しかし，今のところはIntelの最新CPUと同等の性能をもつCPUを多くの会社が販売し，Pentiumやi486の値下がりに拍車をかけてくれることが筆者の切実な願いです．

歴史に残る8ビットパソコン

● **Apple II**

S. WozniakとS. Jobsの作ったApple社が発売したパーソナルコンピュータ．スペックは次のとおり(AppleII)．MPU：6502-1 MHz，RAM：16 Kバイト(48 Kバイトまで拡張可能)，ROM：12 Kバイト，ROMの内訳：6 Kバイト(BASIC)，2 Kバイト(モニター ROM)．モニターROMは，アセンブラ，フローティングポイントパッケージ，スイート16という16ビットMPUのシミュレータが付いていた．ビデオ表示はNTSC出力で最大で280×192ドットの4色表示． (ZigZag)





● **Atari 65XE**

Atari社のコンピュータはマニア受けするものばかりだった．65XEはAtari 800の後継機で，カスタムチップを使用し当時としてはハイスペックを誇っていた．同社のカスタムチップの設計者が後述するAmigaのカスタムチップの設計もしていた．MPU：6502c-1.79 MHz，GTIA：グラフィック専用カスタムチップ，POKEY：音源コントロール専用カスタムチップ，ANTIC：スクリーン，I/O専用カスタムチップ，FREDDY：メモリシステム・コントロール専用カスタムチップ，RAM：64 Kバイト，ROM：24，Kバイト(BASICを含む)，表示：11のグラフィクスモード，256色同時発色，320×192，五つのテキストモード，音源：4声，3 1/2オクターブ． (ZigZag)



● **Commodore 64**

ファミリーコンピュータが世界的に普及する前に世界的に普及していたゲームおよびホーム向けコンピュータ．Apple IIと比べると専用チップの使用が見られ，「今」のコンピュータのようにもみえる．スペックは次のとおり．MPU：6510，RAM：64 Kバイト，CIA(Complex Interface Adaptor)6526×2，VIC - II (Video Interface Controller)6567×1，SID(Sound Interface Device)×1． (ZigZag)







● **MSX2**(標準的構成)

64 Kバイトのメモリ空間を16 Kバイト単位の四つのページに分けたバンク切り替え機構(スロット)を備える国産8ビット機の雄．拡張機器はメモリマップトI/O方式で接続され，スロットにより管理されているので，I/Oアドレスが衝突するなどの問題がない．ただし割り込みがモード1のみなので，割り込み駆動型の拡張機器は負担が大きい．MSX-DOSはMS-DOSとファイル互換．CPU：Z80A相当品-3.58 MHz，RAM：64 Kバイト，ROM：32 Kバイト，サブROM 16 Kバイト，VRAM：128 Kバイト． (M)

● **PC-8801mk2**

フロッピディスク(FDD)ユニット側にもZ80を搭載したインテリジェントFDD．VRAMはC000h以降のアドレス空間をバンク切り替えによりアクセス．後継機はFM音源を搭載したSR，2HD対応のMR，CPUを高速化したFH，16ビットに強化されたVAなどがある．CPU：Z80A相当品-3.58 MHz，RAM：64 Kバイト，ROM：72 Kバイト，拡張ROM 16 Kバイト，VRAM：48 Kバイト，NBASIC． (M)



▲写真は，PC-8001mkII

● **X1 turbo**

クリーンコンピュータ思想，割り込みモード2やDMA搭載など，Z 80ファミリにより固められたCPU周辺回路．またCPUからみるとVRAMはI/O空間にマッピングされている．キーボードから音量やチャネル切り替えなどの操作ができるTVコントロール機能なども標準装備されるなど，そのいくつかの特徴は，X 68000にも受け継がれている．CPU：Z80A，RAM：64 Kバイト，BIOS ROM 32 Kバイト，VRAM：96 Kバイト，HuBASIC，CP/M． (M)



● **FM-7**

FM-7とは略称であり，正式名はFUJITSU MICRO 7という．付属のBASICはF - BASICといい，Microsoftの BASICを同機のために拡張したもの．ホビーからビジネスユースまで幅広く使われた．CRT表示用に専用のマイクロプロセッサ(68B09)を使用し，高速表示が可能であった．MPU：68B09(メインブロック)/68B09(サブブロック用)，RAM：64 Kバイト，ROM：32 Kバイト(BASIC)/2 Kバイト(ブートローダ)，VRAM：48 Kバイト．



歴史に残る68000パソコン

● Amiga

グラフィックスやゲーム，ビデオ編集で知られていたパソコン．しかし，これら以外にも25チャネルのDMA，専用のカスタムチップをもつなどの特徴をもったパソコンだった(図1)．

▶マルチタスク/プロセスOS

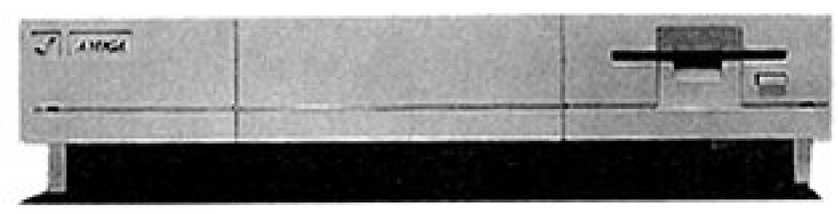
Amigaのはプリエンプティブなマルチタスク/プロセスOS．通信ソフトを動かして，エディタでプログラムを編集し，内部音源で音楽を鳴らし，3Dソフトでレンダリングをし，さらにフロッピディスクをフォーマットすることを，7.16 MHzの68000で行っていた．専用のGUIはWorkbench(Intuition)と呼ばれる．

▶OOPライクなマルチタスクカーネル

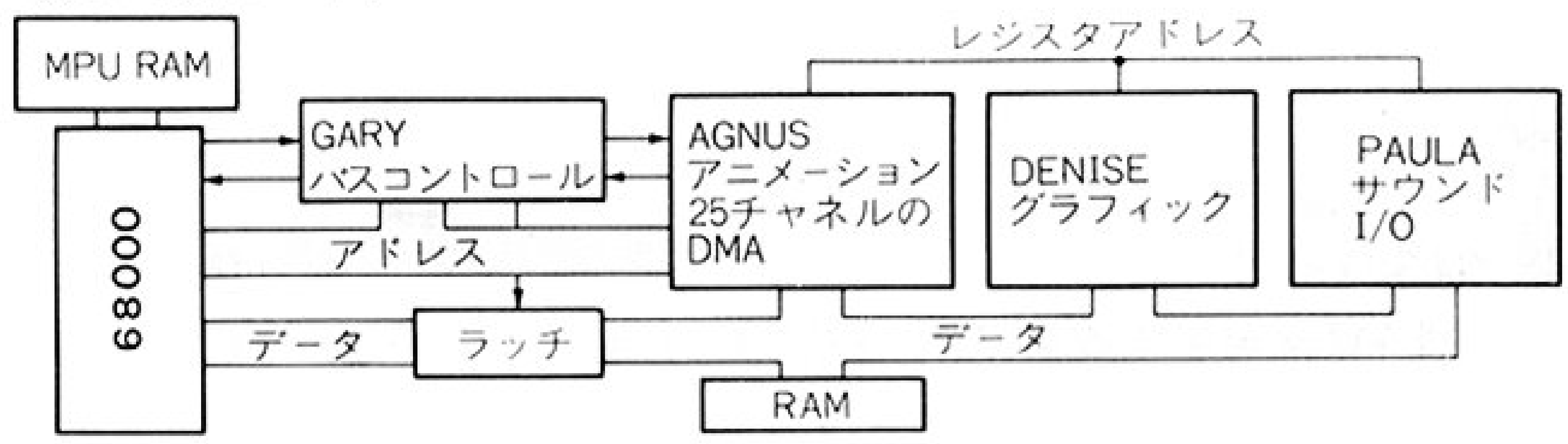
Amigaのマルチタスクカーネルは，なんとオブジェクト指向的な作りになっていた(図2)．また，名前付きパイプによるプロセス間通信もサポートされていた．

▶REXXが標準

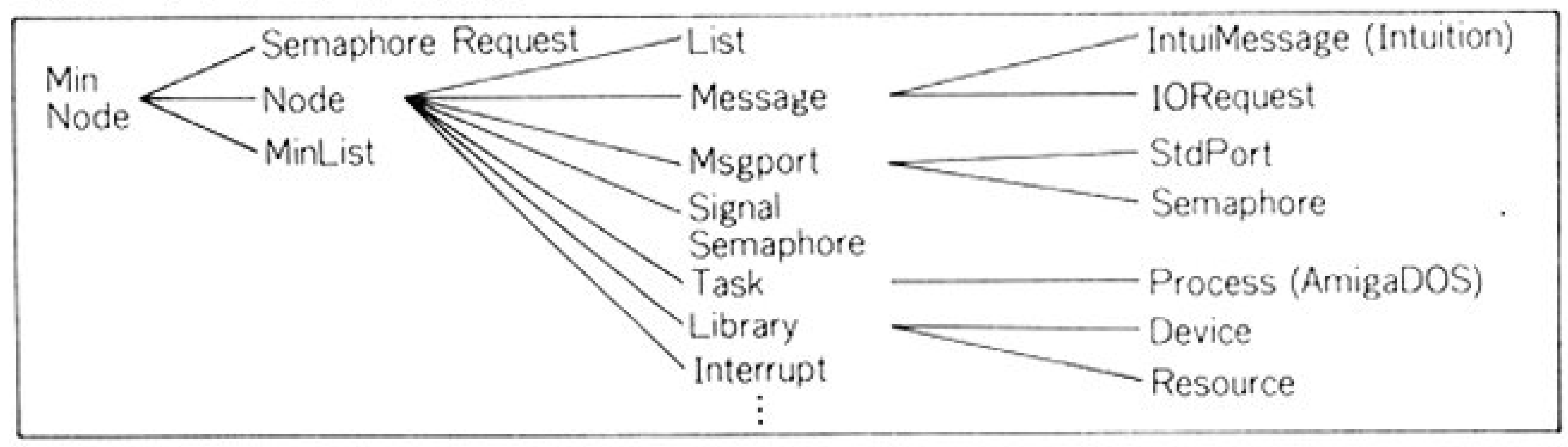
OS/2で知られたREXXだが，Amigaにも標準でついており，コンパイラまで発売されていた． (ZigZag)



〈図1〉内部ブロック



〈図2〉オブジェクトの階層



● Atari ST

ゲームやMIDIで知られたパソコン．

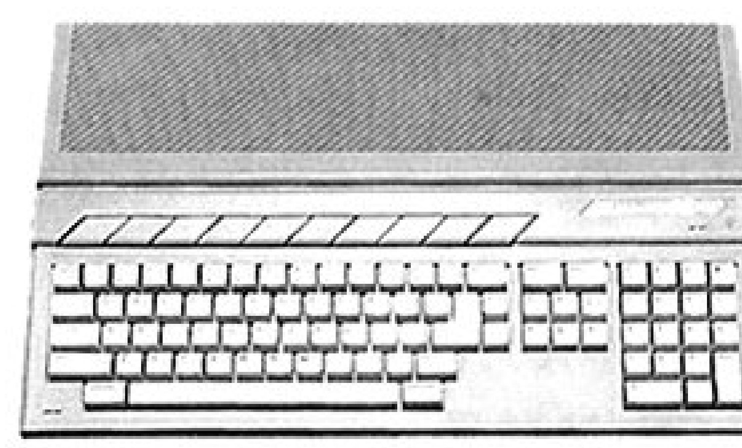
OSは，Digital ResearchのCP/M-68KにGEMというGUIをのせたTOSで，8 MHzの68000で小気味よく動いていた．ソフトも豊富で，特にCubaseといった音楽用ソフトに優れたものがあった．

▶68ファミリで固めたST

STは，68系の周辺LSIで構成された素直な設計である．それは1個の68901と2個の6850を使用していたことからもうかがえる．

▶標準でMIDI

STは，音楽に強いと言われる理由は，優れたソフトだけでなく，MIDIが付属していたことも一因である． (ZigZag)



● Macintosh

Macについては特に語ることもないだろう．今でこそ多くのユーザーに支えられいるMacだが，日本で発売された当初は，まったくユーザーが増えない状況が続いた．アプリケーションもMac-Write，MacPaint以外に目ぼしいものがなく，客観的に見れば当時はMS-DOSパソコンのほうがはるかに「使えた」．

▶初代Macのスペック

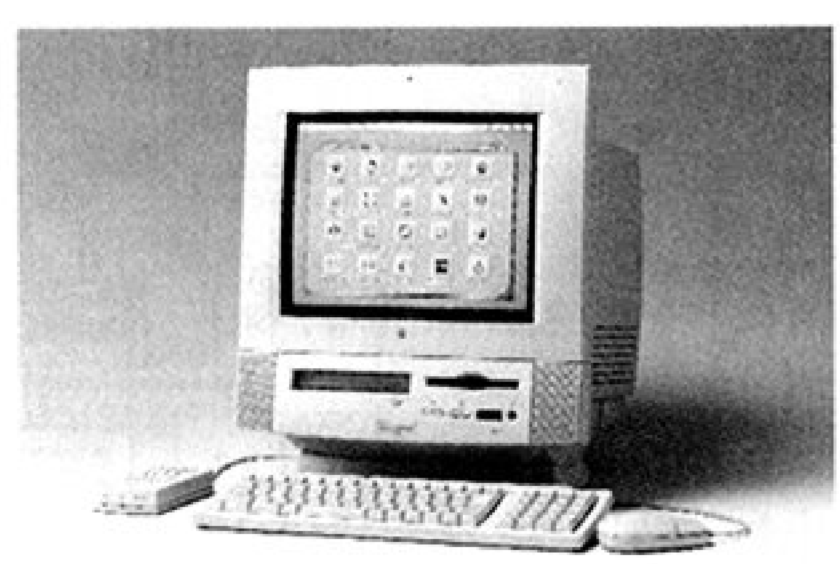
初代Mac 128は，7.8 MHzの68000を使用していながら，メモリがたったの128 Kバイトであり，OSが40 Kバイト以上も占有するので，前述のような状況が続いたのは頷ける．

実際，Macが認知されだしたのはMacPlus，SE，MacIIの時代からで，漢字Talkも次第に安定してきてからである．

また，QuickDRAWを載せたことなどで，よくLisaの後継機といわれるが，システム構成などを見ると別なコンピュータといったほうがよい．識者の間ではLisaの後継はNeXTだと言われている． (ZigZag)



▼68KからPowerPCへ移行している

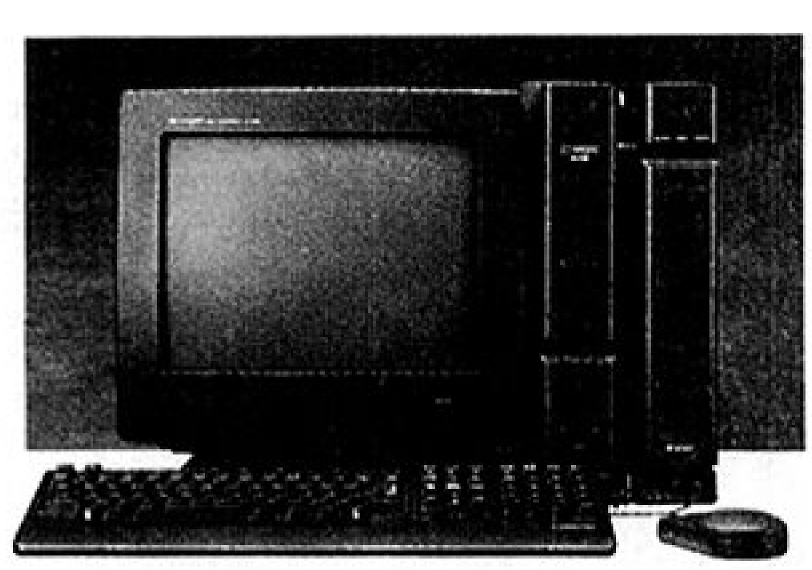


● X68000

1024×1024ドット16色や512×512ドット65536色&スプライトなどの多彩なグラフィックス機能と，FM音源とADPCMのサウンド機能を標準装備したマシン．標準添付のOSであるHuman68kは，MS-DOSライクなシングルタスクOS．Human68k上で動作するSX-WindowというGUIもある．

CPUを68EC030に強化した後継機X68030もある．ここ2年ほどはモデルチェンジの話を聞かないが，一部に熱狂的ファンがついているマシンでもある．

CPU：68000-10 MHz，RAM：2 Mバイト(最大12 Mバイト)，VRAM：512 Kバイト [CZ-623C(X68000 SUPER-HD)]． (M)



MS-DOSとPC用BIOSの歴史

これまで，PCと8086マイクロプロセッサの歴史をみてきましたが，次にMS-DOSの歴史をその登場以前の市場背景から振り返ってみたいと思います．

MS-DOS登場以前

● BASIC時代

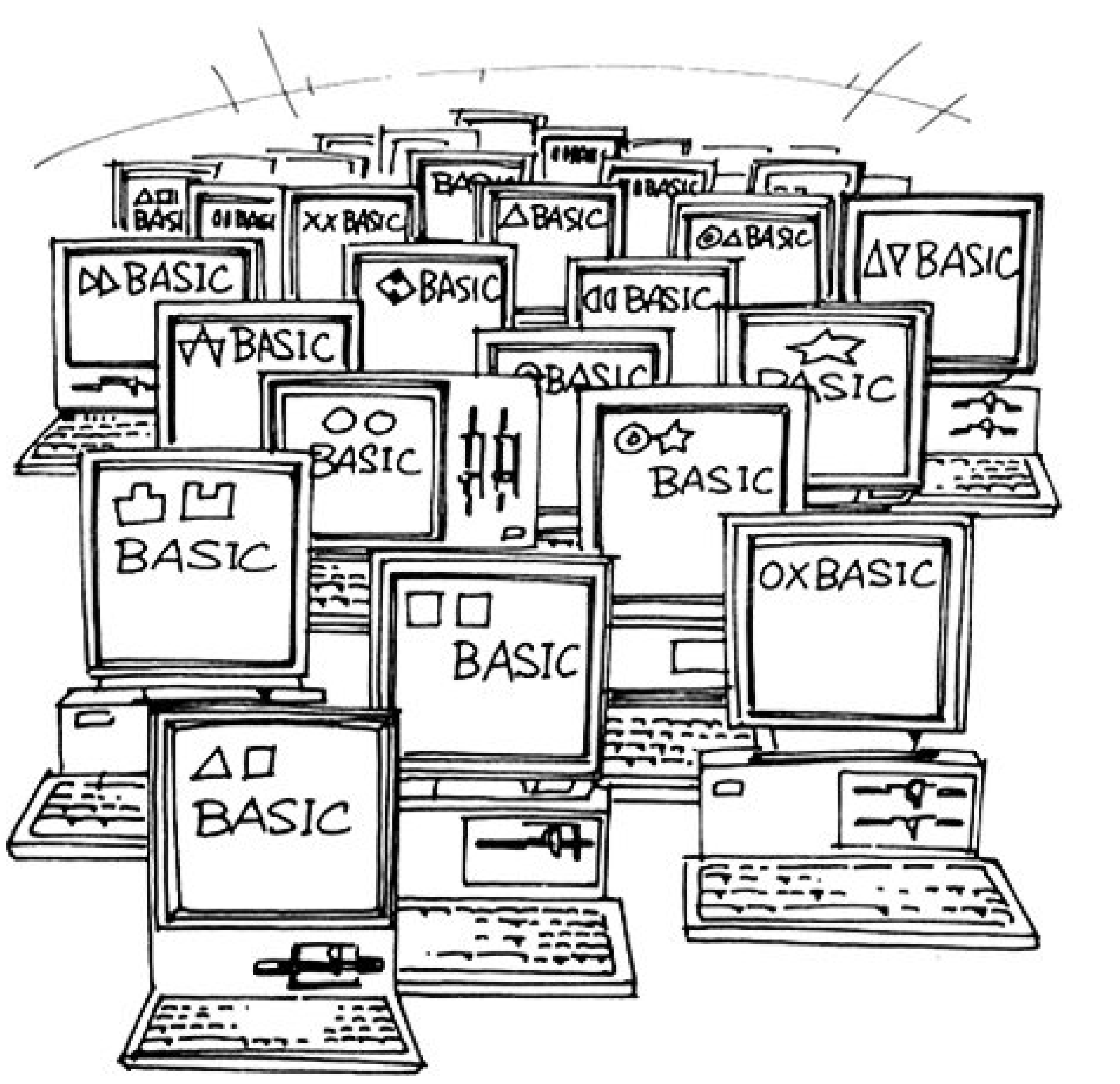
IBM PCが登場するまでは，パソコンの市場はZ80で動作するCP/Mベースのパソコンか，6502を使ったAppleIIに二分されていたと言ってもよいでしょう．

この頃のパソコンはBASICがROMに書き込まれており，スイッチを入れると自動的にBASICインタープリタが動作するようになっていました．このBASICをいろいろなパソコンに移植していた会社の一つがMicrosoftでした．

ただし，それらのBASICも各社のパソコンの仕様に合わせて移植していたため，いろいろな方言(文法や内部表現形式)をもっており，同じZ80系のパソコンであっても互換性はなく，BASICアプリケーションをそれぞれのパソコンごとに作成しなくてはなりませんでした(日本の例を挙げればPC8001のBASICプログラムは，東芝のパソピアでそのままロードしても動かなかった)．

● CP/M時代

その後，Digital Research(現在はNovellに買収された)は，パソコンごとにプログラムを開発しなくてもすむように，画面制御やディスク管理のための統一されたインターフェースと，同じCPUアーキテクチャであれば同一プログラムが実行できるような環境としてCP/Mと呼ぶOSを8080用パソコンのために開発しました．



CP/Mが動作する8080系のパソコンであれば，メーカーや機種が異なっていても一つのプログラムがそのまま実行できるわけですから，アプリケーション開発者にとっては大変ありがたい環境であり，VisiCalc(8ビット時代のExcelみたいなもの)やWordStar(かつては標準的なワープロだった)，MASM(マイクロソフトのかつての本業，マクロアセンブラ)など，この時代を代表する優秀なソフトウェアが次々とCP/M上で動作するようになりました．

このCP/Mとその後に登場するZ80によって，パソコンが一気に趣味やゲームのためだけでなく，スモールビジネスの分野にまで使用されるようになったわけです．ちなみにAppleIIにもCP/Mを走らせるためのZ80カードがありました．

この頃，OSはDigital Research，言語はMicrosoft，アプリケーションはその他という暗黙の棲み分けがなんとなくあったようで，今と比べるとほのぼのとさせられます．

● MS-DOS誕生前夜

さて，IBMは初代PCを開発するにあたって，ハードウェアは自分で開発することを決定していたのですが，基本ソフトウェア(BASICとOS)は他の会社に開発を依頼する方針をとりました．

その結果Microsoftには BASICの開発を依頼し，Digital ResearchにはCP/Mの8086版であるCP/M-86の開発を依頼したのですが，開発スケジュールが初代PCの出荷時期(1981年8月)に間に合わないということで，MicrosoftにOSの開発をも依頼することになります．

というのが表向きの理由ですが，IBMの社員がDigital ResearchのオフィスにOSの開発の依頼に来たときに，Digital Researchの創業者であるゲーリー・キルドールは彼らをすっぽかして？ 飛行機を操縦していたためとか(Digital Researchがあったカリフォルニア州モントレーは誰でもそんな気分になってしまうような観光地です)，IBMの社員がIBM PCプロジェクトに関しての秘密遵守義務を要求したことに彼が納得しなかった，などという話もあります．

ただし，MicrosoftのほうもBASICやその他の言語の開発で手一杯だったので，ビル・ゲイツは別の会社が開発した8086用OSを買い取ることで間に合わせようとします．

そのOSというのは，Seattle Computer Productsという会社が開発した，CP/Mのアプリケーションを





若干手直しするだけで，8086用に動作させせることができるような仕様をもった86-DOSというOSでした．

Microsoftは，この86-DOSの開発者のティム・パターソンを入社させ，若干のバグフィックスをしたのちにMS-DOS(最初はPC-DOSと呼んでいた) Version 1.0として初代PCに組み込むことになります．

MS-DOS Version 1.0 〜CP/M互換8086用OS〜

MS-DOS Version 1.0は，初代PCの64Kバイトのメモリのうち12Kバイトを使用し，160Kバイトの5インチディスクをサポートしていました．

MS-DOS自身はアセンブラで4000行のソースファイルからなり，CP/Mと同様のファンクションコールを備えていたため，8080用に書かれていたアプリケーションは若干の修正をして8086用にアセンブルし直せばMS-DOS上で動作するようになっていました．

この時期のファイルシステムはCP/Mと同様に階層構造をもっていませんでしたが，バッチファイルの機能やシェルとしての**command.com**はすでにこの頃からありました．

なお，PCはハードウェアを操作するファンクションをBIOS(Basic I/O System)としてROM化し，ハードウェアの違いにかかわらずソフトウェアインターフェースを統一するようにしており，MS-DOSもBIOSをコールしてI/Oを操作するようになっていました．

このことと，PCの回路図やBIOSのソースリストが公開されていたことにより，サードベンダーはPCと同一のハードウェアとBIOSを作成すればMS-DOSがそのまま動作するPC互換機を作ることができました．

これが，結局はIntelとMicrosoftだけが大儲けして，IBMだけ大損してしまうというPC市場になった最大の原因です．

なお，Microsoftはこの後の1982年10月にバグフィックスと320Kバイトのフロッピディスクに対応したVersion 1.1を出荷します．

ただし，Digital Researchも後からPC用のCP/M-86を出荷することになるのですが，技術的に両社の違いはほとんどなく，どちらのOSが主流になるのか不透明な時期でした．

しかし結局は，PCを初めから買うと付属してくるMS-DOSとオプションで値段が高いCP/M-86というマーケティング的な戦略にCP/M-86は苦境に立たされます．

MS-DOS Version 2.0 〜Unixの思想を取り入れた〜

IBMが1983年3月にPC/XTをリリースしたときに，MS-DOSも新しいバージョン(Version 2.0)が付いてきました．

Version 1.0はCP/M-80のアプリケーションをスムーズにPCに移植させることにありましたから，CP/Mとの違いはそれほどありませんでしたが，Version 2.0からはUnixの思想を取り入れたいろいろな拡張がなされています．

CP/Mと同様なファイル関係のシステムコールはそのまま残っていますが，新しくUnix風の**OPEN**, **READ**, **WRITE**, **CLOSE**などのファンクションコールとファイルディスクリプタによるファイル操作が加わりました．

シェルの機能も強化され，パイプやフィルタが使えるようになり，ディスクも360Kフォーマットが追加



されています．

Version 2.0のソースファイルはアセンブラで2万行にも達し，これらの機能追加によってCP/M-86との優位性が歴然となり，MS-DOSが完全にPCの標準OSとみなされるようになりました．

またMicrosoftは，この頃から多国語対応のMS-DOSの開発に着手し，何種類かのマイナーチェンジのあとに出てきたMS-DOS Version 2.11がミリオンセラーとなりました．

MS-DOS Version 3.0 〜ネットワークへの対応〜

1984年8月に登場したPC/ATとともにMS-DOSのバージョンも3.0となります．

PC/ATは80286を搭載したPCですから，80286のプロテクトモードを使用すればマルチタスクやリングプロテクションなどの高機能を実現できるPCなのですが，MS-DOSはそのような機能への対応をしませんでした．

そのため，PC/ATでMS-DOSを使用する場合には単なる高速な8086として動作するだけだったわけです(その後，サードパーティからいろいろなDOS用のユーティリティが出てきます)．

ただし，シェルをOSから切り離し，その代わりにユーザープログラムを自動的に起動させたりする機能が追加されています．

ソースコードもアセンブラで4万行となり，Microsoftの開発スタッフも30人になっていました．

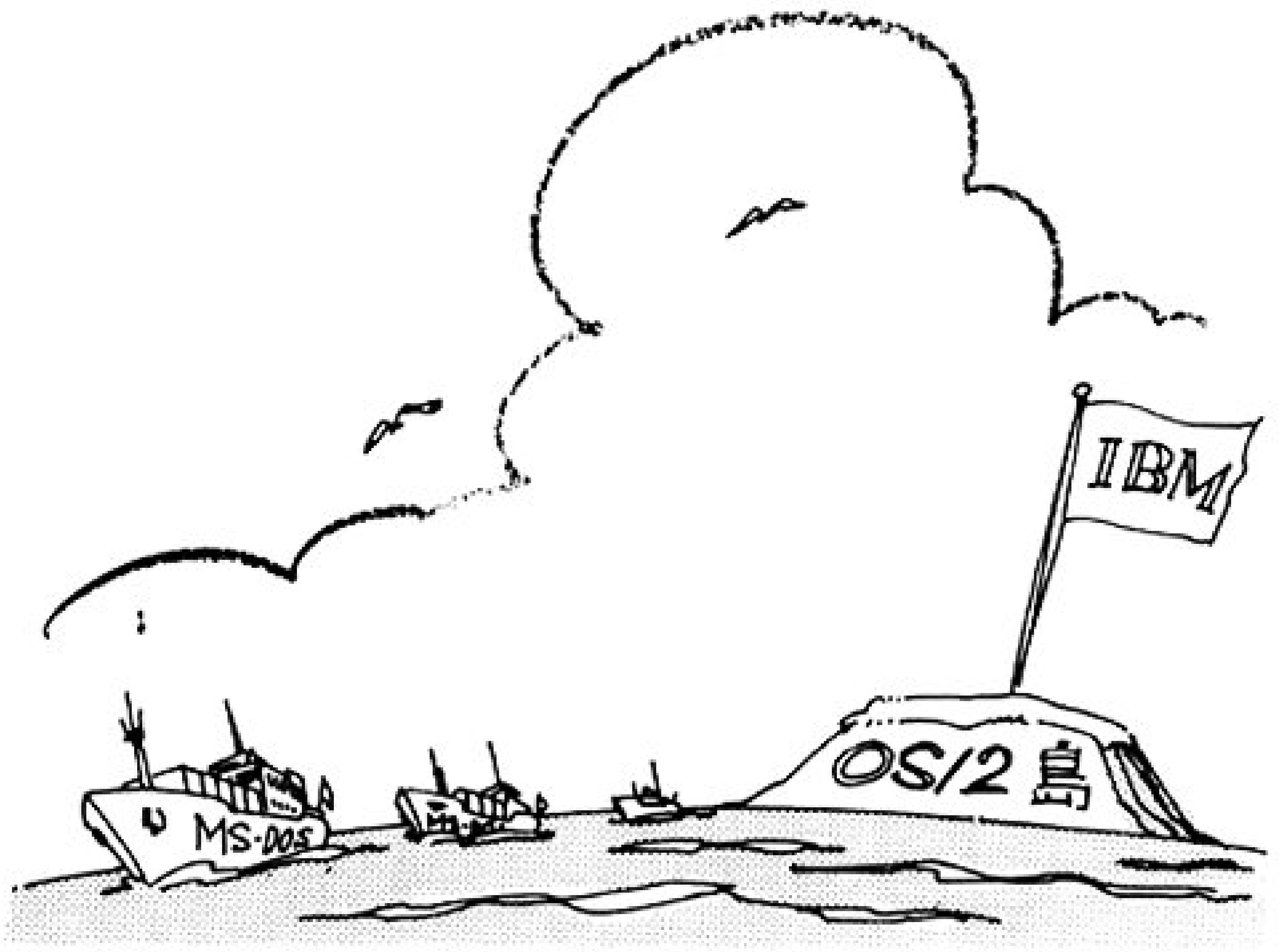
Version 3.0は1984年11月にVersion 3.1にアップグレードされましたが，このバージョンで初めてMS-DOSでネットワークの対応を図っています(が，ほとんど使いものになる出来ではなかった)．

マイナーなバージョンでトークンリングと3.5インチディスクをサポートしたVersion 3.2はバグの塊で，ほとんどのユーザーは3.1に戻るか，次のメジャーなバージョンアップを待つことになります．

1987年にIBMはそれまでのPCからPS/2へと路線を変更し，MS-DOSもVersion 3.3が搭載されました．

IBMとMicrosoftはこの時点でOS/2の開発を発表し，いろいろな利点をもつOS/2にMS-DOSを置き換える方針を決定しました．

しかし，OS/2の出荷の遅れとMS-DOSとの互換性の低さが響いてMS-DOSの置き換えをすることはできず，1991年にMicrosoftはIBMと共同でOS/2を開発する契約を解消することになります．



MS-DOS Version 4.0 〜大容量ハードディスクへの対応〜

OS/2の市場への普及が(1回目の)失敗に終わった後に，IBMをさらに驚かす現象がVersion 4.0によってもたらされたのはIBMにとっては皮肉なことです．

MS-DOSを市場からなくそうとして行ったいろいろな方策が，結果としてMS-DOSの拡張を促す結果になっただけでなく，IBMが完全にPC市場から脱落する結果をもたらすことにもなったわけですから．

Version 4.0は，その頃に普及していたPC互換機のために作成されたバージョンで，Version 3.3で問題となっていたハードディスクの最大容量を32Mバイトから最大2Gバイトに拡大し，プロテクトモードでしかアクセスできない1Mバイト以降のメモリをRAMディスクとして使用可能にしたり，DOSShellを追加したりしています．

ただし，このバージョンはあまり普及せず，次のVersion 5.0で大きな発展をみることになります．

MS-DOS Version 5.0 〜メモリマネージャの追加〜

Version 5.0は1991年の4月に発表されたバージョンで，80286やi386で深刻な問題となっていた1Mバイト以降のメモリの有効活用を図った最初のバージョンです．

640Kバイトの壁は依然として存在していましたがMS-DOS自身をHIMEM部分に移動させたり，デバ

〈写真18〉
IBM DOS J5.0/V



〈写真19〉
MS-DOS 6.0



(a) MS-DOS 6



(b) MS-DOS 6.2/V

イスドライバをUMBに移行させたりして，640Kバイトのメインメモリ内の600Kバイト程度をフリーエリアにすることができるようになりました.

また,MS-DOSのアプリケーションを複数常駐させて切り替えることで，マルチタスクを行うことが可能なDOSShellもサポートされています.

出荷におけるテストもより完璧なものとなり，それまでのMS-DOSに比較すると非常に安定したバージョンでもあります.

また，このバージョンからMS-DOSは一般のパソコンショップでMicrosoft製品として発売されました(米国市場の場合)．日本では，DOS/Vがこのバージョンでサポートされたことは記憶に新しいところです(**写真18**).

MS-DOS Version 6.0 ～ユーティリティの付属～

このバージョンでは，特にMS-DOSそのものにおける機能拡張は目立ったものはありませんでしたが，サードパーティが開発し，爆発的なヒットとなったディスク圧縮ユーティリティ(Stacker)やメモリ管理ソフト(QEMM386)などと同等の機能として，Double SpaceとMEMMAKERをMS-DOSに添付したことと，評判のよかった他社製のMS-DOSおよびMS-Windows用のユーティリティを買い取ってMS-DOSのパッケージに追加したことでしょう(**写真19**).

中でもディスク圧縮ユーティリティは，6.0で採用されたDouble Spaceにバグが多数発見され，そのためにその当時のスタンダード的な製品であったStack ElectronicsのStackerとの間で訴訟を起こして裁判で争うまでに発展しました.

両社ともそれぞれの主張がそれなりに認められましたが，MicrosoftがStackerの技術を盗用していると判断されたり，ディクス圧縮ツールのDouble SpaceのバグフィックスのためにVersion 6.2が出ました.
さらに，その後でディスク圧縮ツールをDrive Spaceという新しいものに取り替えるために6.22が出たりしたため，筆者はディスク圧縮ツールを使っていないにもかかわらず，1994年に3回もMS-DOSをアップグレードしてしまいました(必要はないのだが,安かったからそうしただけです).

なお，最近のMicrosoftは裁判沙汰になることも増え，自社製品が太刀打ちできない優れたツールを出しているようなベンチャ企業を買収することも多くなっています.

BIOSをめぐる歴史

BIOSの歴史は，PC互換機の歴史そのものであるといってよいでしょう.

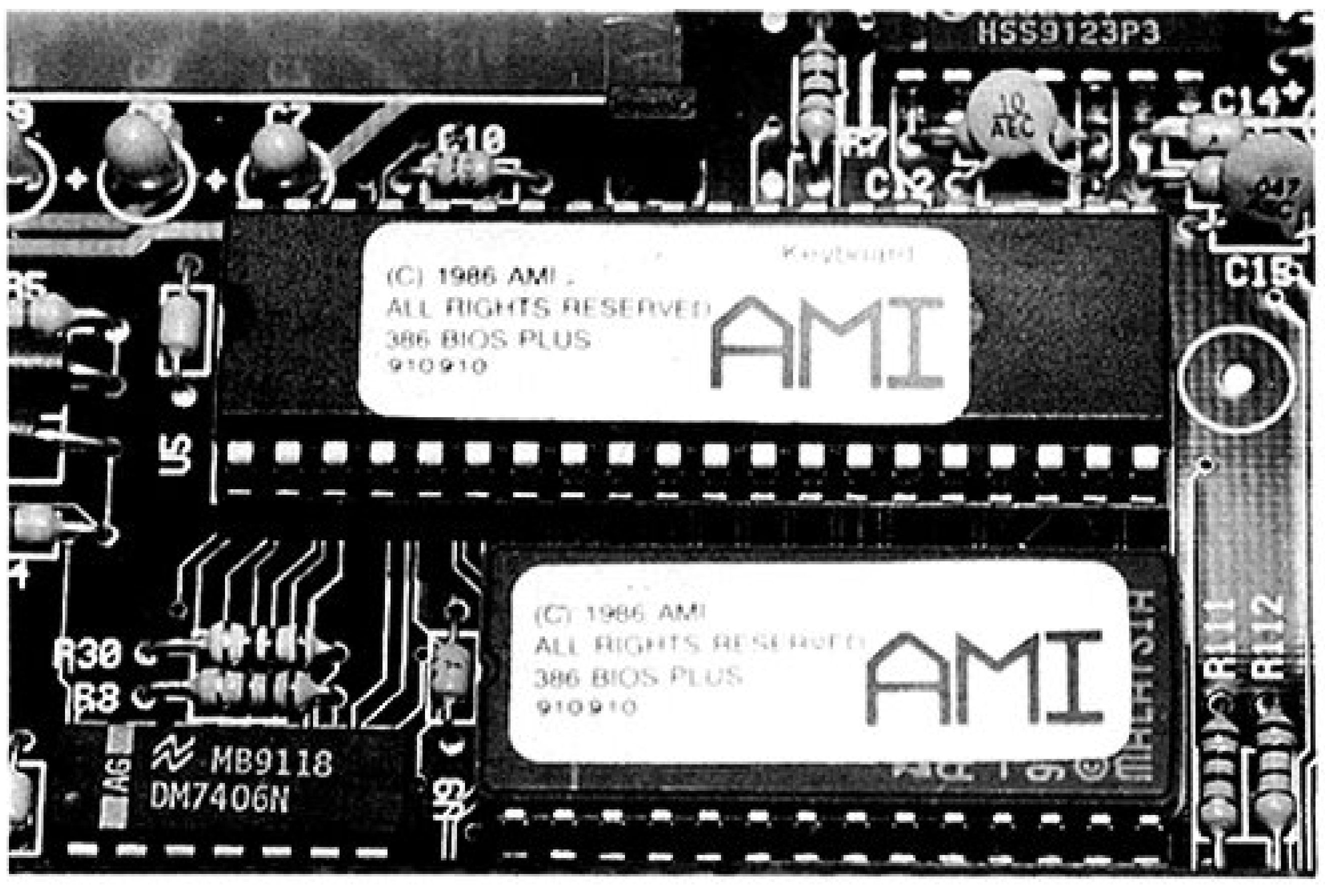
かつて，IBMだけがPCを出荷していたときには，BIOSはIBMが作成したものだけがソースリストとともに存在していたわけですが，互換機メーカーは機能とソフトウェアインターフェースがIBMのBIOS

```
 IBM Personal Computer MACRO Assembler   Version 2.00            1-12
TEST1 ---- 11/15/85  POWER ON SELF TEST (POST)                    11-19-

889                              PAGE
890                              ;-----  CHECK FOR SHUTDOWN 09
891  0091                        C7B:
892  0091 B0 8F                          MOV     AL,CMOS_SHUT_DOWN+NMI    ;
893  0093 E6 70                          OUT     CMOS_PORT,AL
894  0095 EB 00                          JMP     $+2                      ;
895  0097 E4 71                          IN      AL,CMOS_DATA             ;
896  0099 3C 09                          CMP     AL,09H                   ;
897  009B 86 C4                          XCHG    AL,AH                    ;
898  009D 74 41                          JE      C7C                      ;
899
900                              ;-----  CHECK FOR SHUTDOWN 0A
901
902  009F 80 FC 0A                       CMP
903  00A2 74 3C                                                           ;
904
905  00A4 2A C0                                                           ;
906                                              X287+1,AL

                                 ;--------------------------------------
                                 ;   RE-INITIALIZE THE 8259 INTERRUPT #1 C
                                 ;--------------------------------------
911  00A8 B0 11                          MOV     AL,11H                   ;
912  00AA E6 20                          OUT     INTA00,AL
913  00AC EB 00                          JMP     $+2                      ;
914  00AE B0 08                          MOV     AL,08H                   ;
915  00B0 E6 21                          OUT     INTA01,AL
916  00B2 EB 00                          JMP     $+2                      ;
917  00B4 B0 04                          MOV     AL,04H                   ;
918  00B6 E6 21                          OUT     INTA01,AL                ;
919  00B8 EB 00                          JMP     $+2                      ;
920  00BA B0 01                          MOV     AL,01H                   ;
921  00BC E6 21                          OUT     INTA01,AL                ;
922  00BE EB 00                          JMP     $+2                      ;
923  00C0 B0 FF                          MOV     AL,0FFH                  ;
924  00C2 E6 21                          OUT     INTA01,AL                ;
925                              ;--------------------------------------
926                              ;   RE-INITIALIZE THE 8259 INTERRUPT #2 C
927                              ;--------------------------------------
928  00C4 B0 11                          MOV     AL,11H                   ;
```

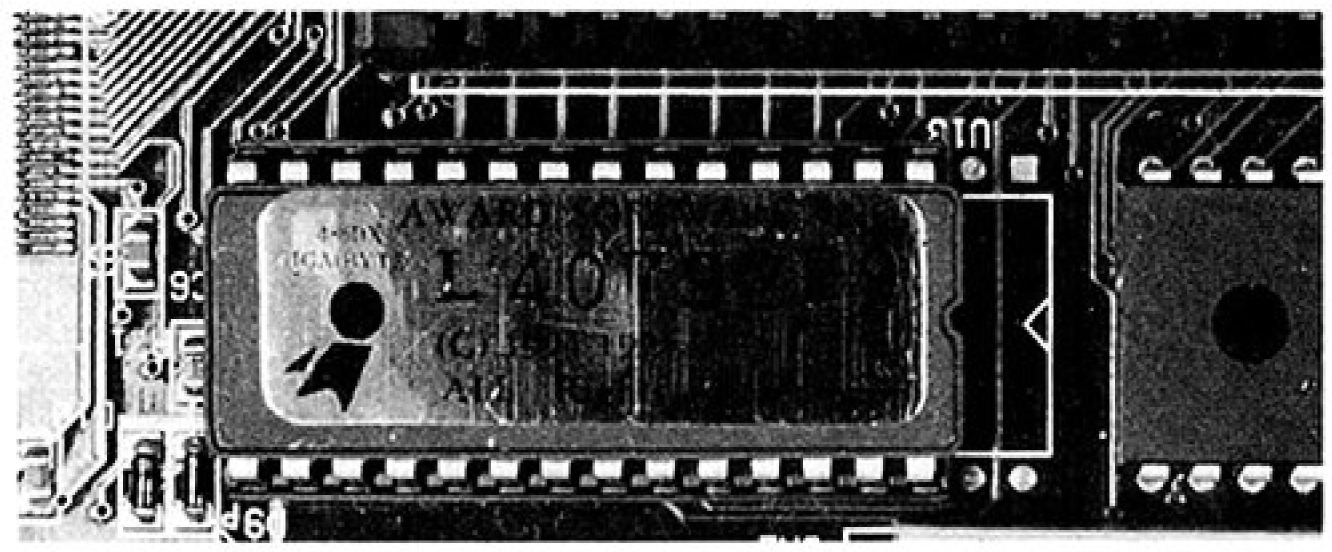
〈写真20〉 AT BIOSのソースの一部



(a) 386 BIOS (AMI)



(b) XTBIOS (Award)



(c) 486 BIOS (Award)

〈写真21〉 各社の BIOS

と同じものを作成し，互換機のマザーボードに ROM 化して出荷し始めました(**写真20**)．

初めの頃は BIOS の互換性も 100%とはいえなかったのですが，PC/AT 互換機が普及し，IBM が PC 市場におけるシェアを失う頃には BIOS の互換性を論議する必要がないほど互換性が高くなり，逆に互換性を計る標準機として Compaq の PC 互換機などが使用されるようになりました．

なお，かつては BIOS をめぐる法廷での論争も盛んでしたが，「ある特定の目的を達成するプログラムのソフトウェアインターフェースが他の同様なプログラムと同じになっていても，それは著作権の侵害にはならない」という判例が出たことにより，BIOS における法的問題はすでに過去のものになっています．

今のところ，BIOS を他の互換機メーカーに供給しているベンダーとしては AMI，Pheonix，Award などがあります(**写真21**)．

どんどん集積度が高まって色々な機能が追加されていくチップセット(後述)のサポート(初期設定やコンフィグレーション)を BIOS で対応する必要もあり，この 3 社がほぼ BIOS 市場を独占している状態です．

一般に入手できる互換機のマザーボードには，ほとんどの製品にこれらの BIOS が使われています(ただし，IBM や Compaq は自分の PC 用に BIOS を作っている)．

また，最近では省エネルギを考慮した EnergyStar 対応の BIOS や，Plug and Play に完全対応させるために，BIOS ROM にプログラムされていた I/O の割り込みやアドレスの設定をソフトウェアで書き換えられるようにした FLASH BIOS なども出てきています(すなわち，BIOS がフラッシュメモリにプログラムされており，アドレスや割り込みの設定が書かれている BIOS の部分を後からソフトウェアで書き直すことができる)．

バスとチップセットの変遷

PC のバス，いわゆる拡張スロットの発展の歴史は高速化する CPU に対応するためと，PC 市場をめぐる各社各様の思惑が入り乱れて，様々なバスが登場しています(**表 5**)

XT バス

初代 PC，および PC/XT の頃は，拡張スロットは単に CPU のバスをバッファリングした信号線を持っていたにすぎませんでした．

当時の PC は，このスロットに増設メモリを付けたりしていましたが，CPU は 8088 の 4.77 MHz でしたし，転送スピードも 2.38 M バイト/sec しかなくても，スピードの点で不利になることはまったくありませんでした．

前述のとおり，初代 PC では回路図が公開されていましたので，バスのタイミングなどの特性は定義されず，ボードベンダーはボードを開発するときに，8088

〈表5〉バスの変遷

バス	アクセス方法	バンド幅	基本クロック
AT	メモリアクセス1ウェイト	4.0 M バイト/sec	6.0 MHz
AT	メモリアクセス2ウェイト	1.5 M バイト/sec	6.0 MHz
AT	メモリアクセス1ウェイト	5.3 M バイト/sec	8.0 MHz
AT	メモリアクセス2ウェイト	2.0 M バイト/sec	8.0 MHz
AT	プログラム I/O	2.2 M バイト/sec	6.0 MHz
AT	プログラム I/O	2.9 M バイト/sec	8.0 MHz
EISA	メモリアクセス(バースト)	33 M バイト/sec	8.33 MHz
EISA	メモリアクセス(平均)	24 M バイト/sec	8.33 MHz
EISA	DMA	2 M バイト/sec	8.33 MHz
EISA	DMA タイプ A 16 ビット	2.6 M バイト/sec	8.33 MHz
EISA	DMA タイプ A 32 ビット	5.3 M バイト/sec	8.33 MHz
EISA	DMA タイプ B 16 ビット	4.0 M バイト/sec	8.33 MHz
EISA	DMA タイプ B 32 ビット	8.0 M バイト/sec	8.33 MHz
EISA	DMA タイプ C 16 ビット	16.6 M バイト/sec	8.33 MHz
EISA	DMA タイプ C 32 ビット	33.0 M バイト/sec	8.33 MHz
DRAM バンク	32 ビットページモード	44 M バイト/sec	33 MHz
VL ローカルバス	32 ビットアクセス	66 M バイト/sec	33 MHz
PCI ローカルバス	32 ビットアクセス	120 M バイト/sec	33 MHz
'486 ローカルバス	32 ビットメモリアクセス(バースト)	133 M バイト/sec	50 MHz

のバスを TTL でバッファリングしたものとしてタイミング計算を行っていました.

AT バス(ISA)

AT バスは PC/AT のときに採用されたバスです. PC/AT は 8 MHz の 80286 を使用しているため, アドレス信号線を 24 ビット, データバスを 16 ビットに拡張するだけではなく, タイミングも 8 MHz の CPU クロックに合わせて高速化されています.

信号線の拡張のために, XT バスのコネクタの隣に直線的に並べた新たなコネクタを設けています.

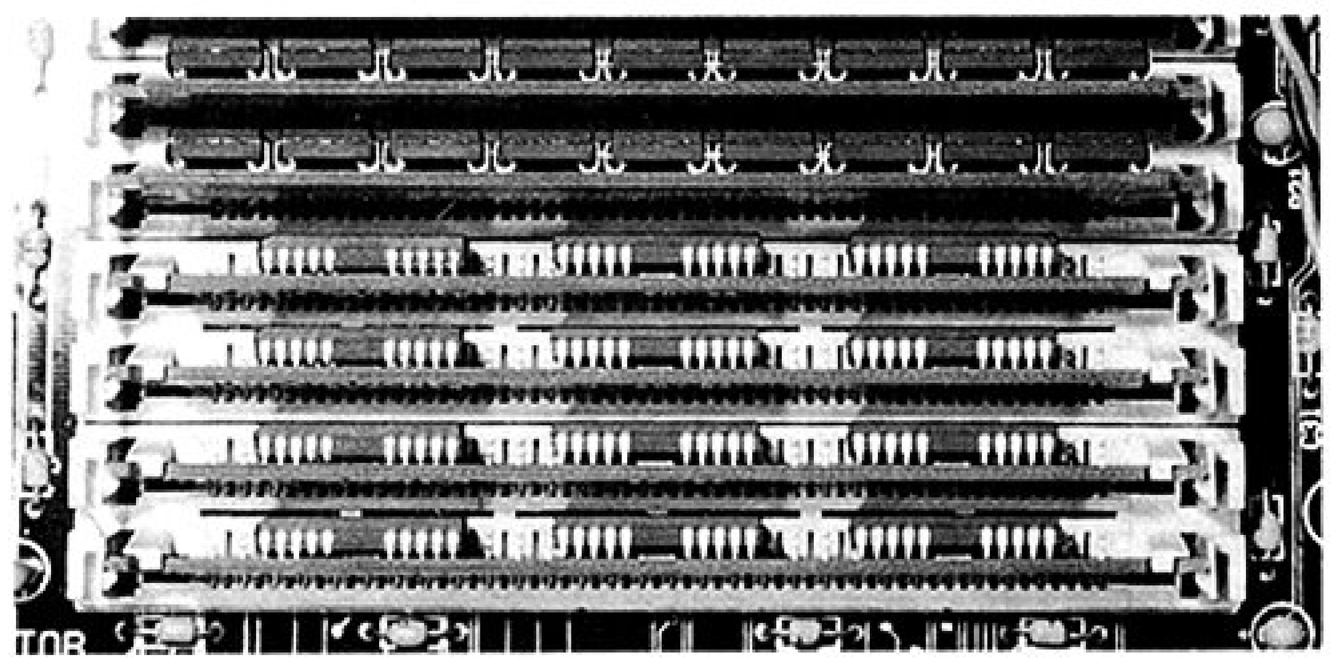
ピーク時の転送スピードは 8 M バイト/sec になったのですが, ここでも IBM は AT バスのタイミング特性を公開しませんでしたので, ボードベンダーは不明なタイミング特性の部分を予想や実測で求めなくてはなりませんでした.

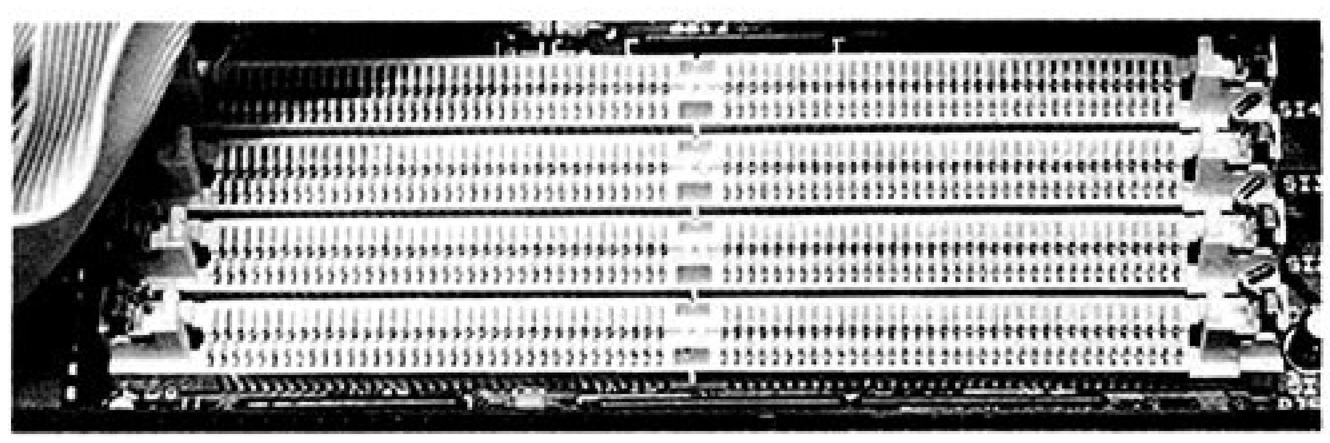
また, AT 互換機の登場により, バススピードよりもはるかに高速な CPU が使用されるようになり, バスのタイミング特性を定義する必要がいっそう高まってきました.

そのため IEEE は, 1987 年に Industory Standard Architecture(ISA)という名称で正式に AT バスのタイミング特性を定義したわけです.

また, 1993 年 5 月には Intel と Microsoft が主体となって Plug and Play ISA Specification も導入されています.

なお, PC/AT 互換機の進化により i386 以降の互換機では, CPU のクロックが加速度的に高速になりました. そのため 8 MHz を基準とする AT バスでは決定的なボトルネックとなりました.


(a) SIMM メモリの装着状態(30 ピン)


(b) SIMM ソケット(72 ピン)

〈写真22〉SIMM メモリとソケット

従来の I/O ボードなどはそのまま使用できるのですが, 拡張メモリなどはスロットに差してもアクセスにウェイトがかなり入ってしまうため, この頃のマザーボードからはメモリ増設専用バスを設けることが多くなりました(今では SIMM のソケットになっている, **写真22**).

なお, 現在の PC マザーボードは AT バスの動作クロックを 8 MHz よりも高速にすることができるものが多く, BIOS のセットアップでチップセットのコンフィグレーションを変更することにより, CPU クロックを分周したものが使えます.

現在発売されているISA用のボードは8 MHz以上のクロックでも正しく動作するものがほとんどですから，この設定を変更することによりシステムのパフォーマンスを若干改善することができます．

MCA (Micro Channel Architecture)

ATバスの問題点を新規設計で解決しようとしたのがPS/2で採用されたMCAです．

MCAにはいろいろと改善点があるのですが，前述のとおり市場からは，ほぼ消え去っています(使っている人ごめんなさい)．

EISA(Enhanced ISA)バス

PS/2に対するIBMの情報非公開(ロイヤリティ制度)に対抗するため，1988年の9月にPC/AT互換機メーカー("Gang of Nine" と当時呼ばれていた)，AST，Compaq，Epson，HP，NEC，Olibetti，Tandy，Wise，そしてZenithの9社がEISAバスの仕様を開発し決定しました．

MCAはATバスの置き換えとして開発されましたが，EISAはその名のとおり従来のISAを拡張して，バスを32ビットにしたりバススピードを高速にしたりしたもので，従来のISAボード用のカードもそのまま差せるような構造のコネクタを使用しています．

転送レートは最大で33 Mバイト/secまで向上し，次に登場したEISA-2では132 Mバイト/secの転送レートを得ることができます．

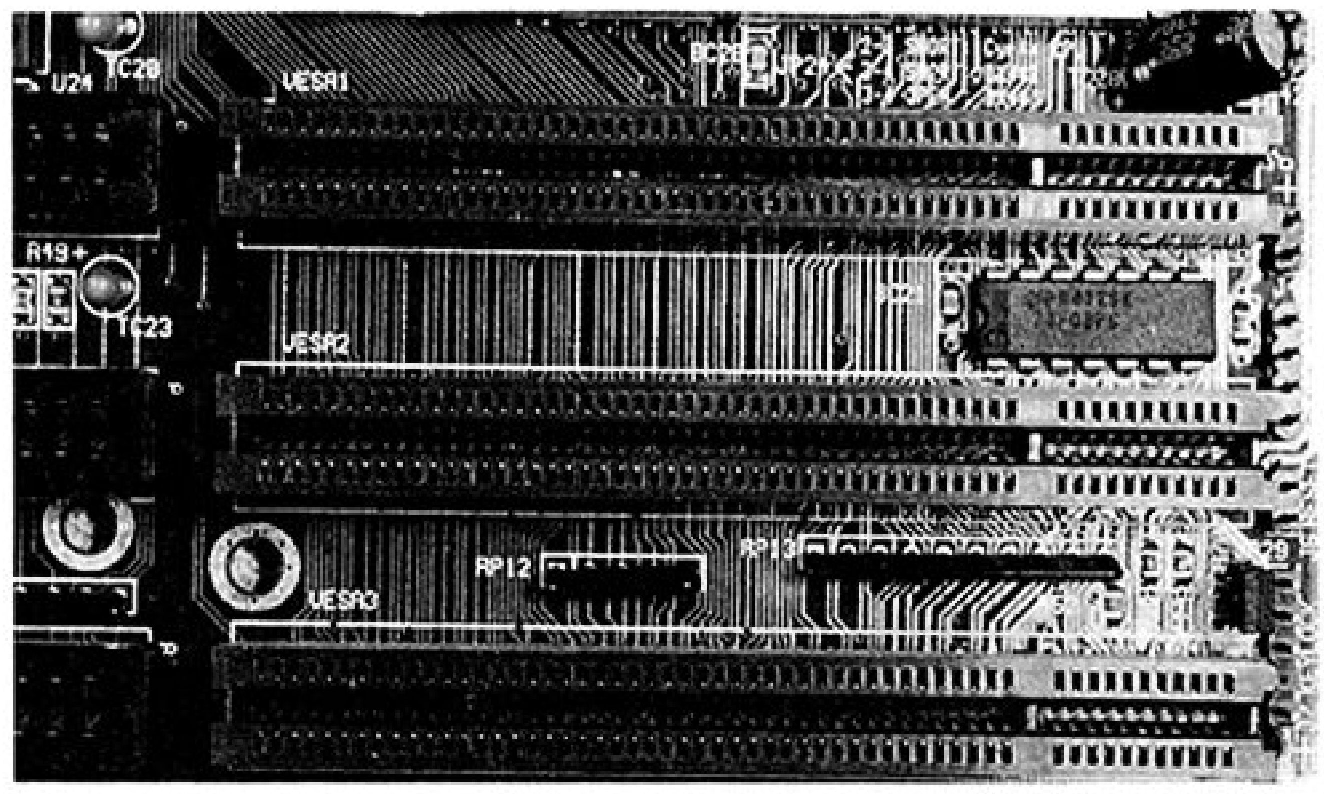
今でもEISAシステムやEISA対応のボードは現役で使用/発売されているのですが，これもしばらくすると後述するPCIに全面的に置き換わることになるでしょう．

いろいろなローカルバスの登場

MCAとEISAの登場により，市場はこの二つのバスの争いになるかと思われましたが，両者とも予想以上に普及が遅れてしまいました．

MCAはIBMによる政治的な配慮があだになったのですが，技術的には新しいバスに対応したボードの価格と品揃えがなかなかユーザーの満足する段階にいたらなかったことと，両社とも「スイッチフリー」，つまり割り込みやI/Oアドレスの設定をジャンパやスイッチではなく，ソフトウェアで行おうとしたために，複雑なソフトウェアセットアップ手順が必要になり，ユーザーにあまり受け入れられなかったためです．

そのため，チップセットメーカーや互換機メーカーは従来のISAの縦に並ぶコネクタを設けて，ISAバスのボードと同じような使用方法で高速なバス転送が行えるバスをいろいろと独自に開発し始めました(これで三列縦隊となる)．



〈写真23〉 VLバス

時代背景としては，Windowsの普及によるグラフィックデータの高速転送が急務となっていた頃です．

これらのバスは，単にバスの転送レートが高速であればよいわけで，CPUの信号線をそのまま接続していたり，アービトレーションや割り込みはISAバスと同じものを使用していましたので，ローカルバスと呼ばれるようになりました．

ただし，1992年ぐらいまで互換機メーカーそれぞれ固有のローカルバスを使用していたのですが，その後に登場するVESAローカルバスが標準的なローカルバスとなり，それからやや遅れてIntelが開発したPCIが普及し始めています．

VESA(Video Electronics Standards Association)ローカルバス，VLバス

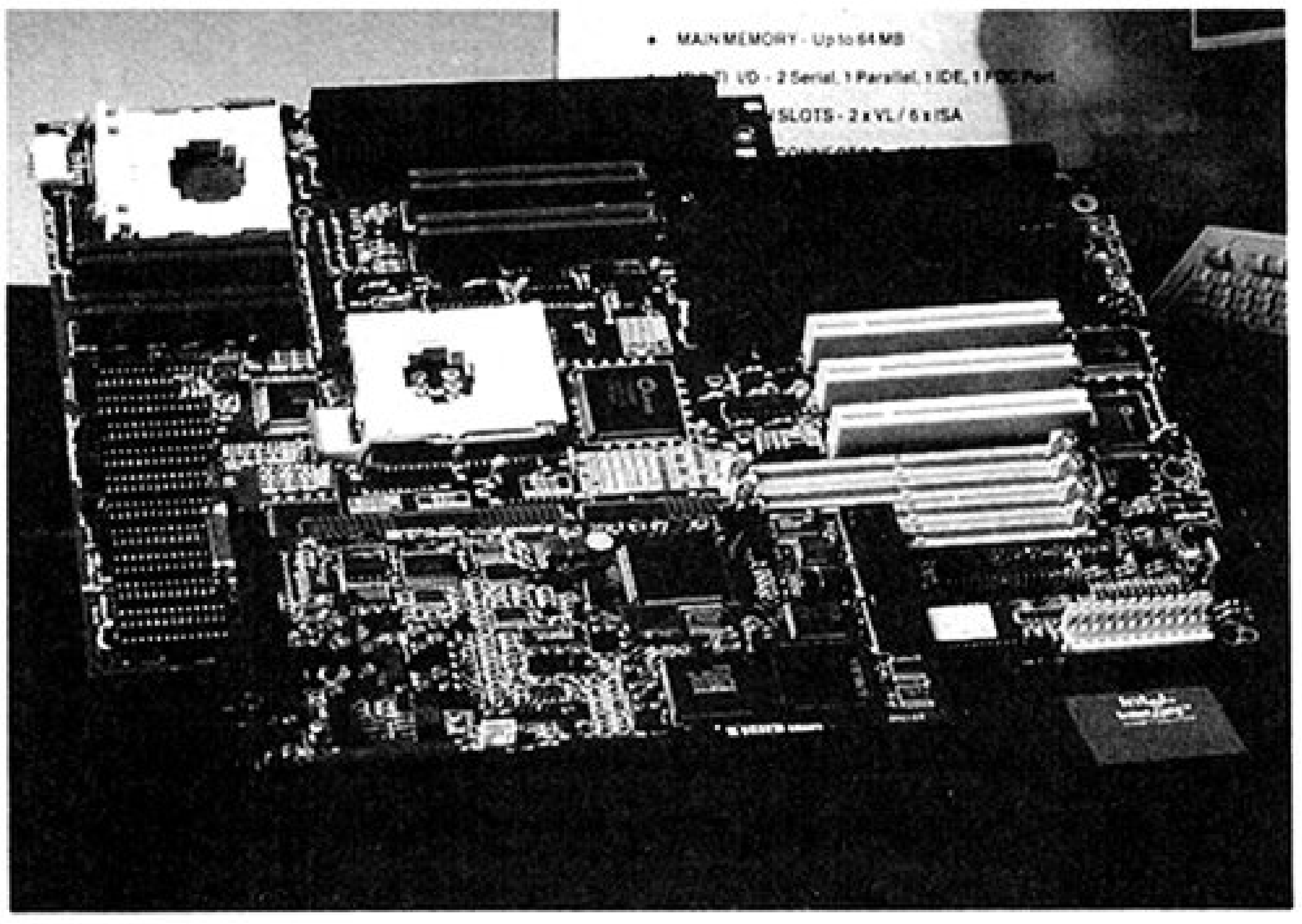
1992年8月に正式に登場したVLバスは，まさにPC業界の望むバスでした(**写真23**)．

標準的なコネクタと使用方法でバススピードを改善するVLバスは，当時のハイエンドPCの致命的ボトルネックだった画像データの高速転送をもたらしただけではなく，ハードディスクやネットワークボードまでVLバス対応のものが使用されるようになりました．

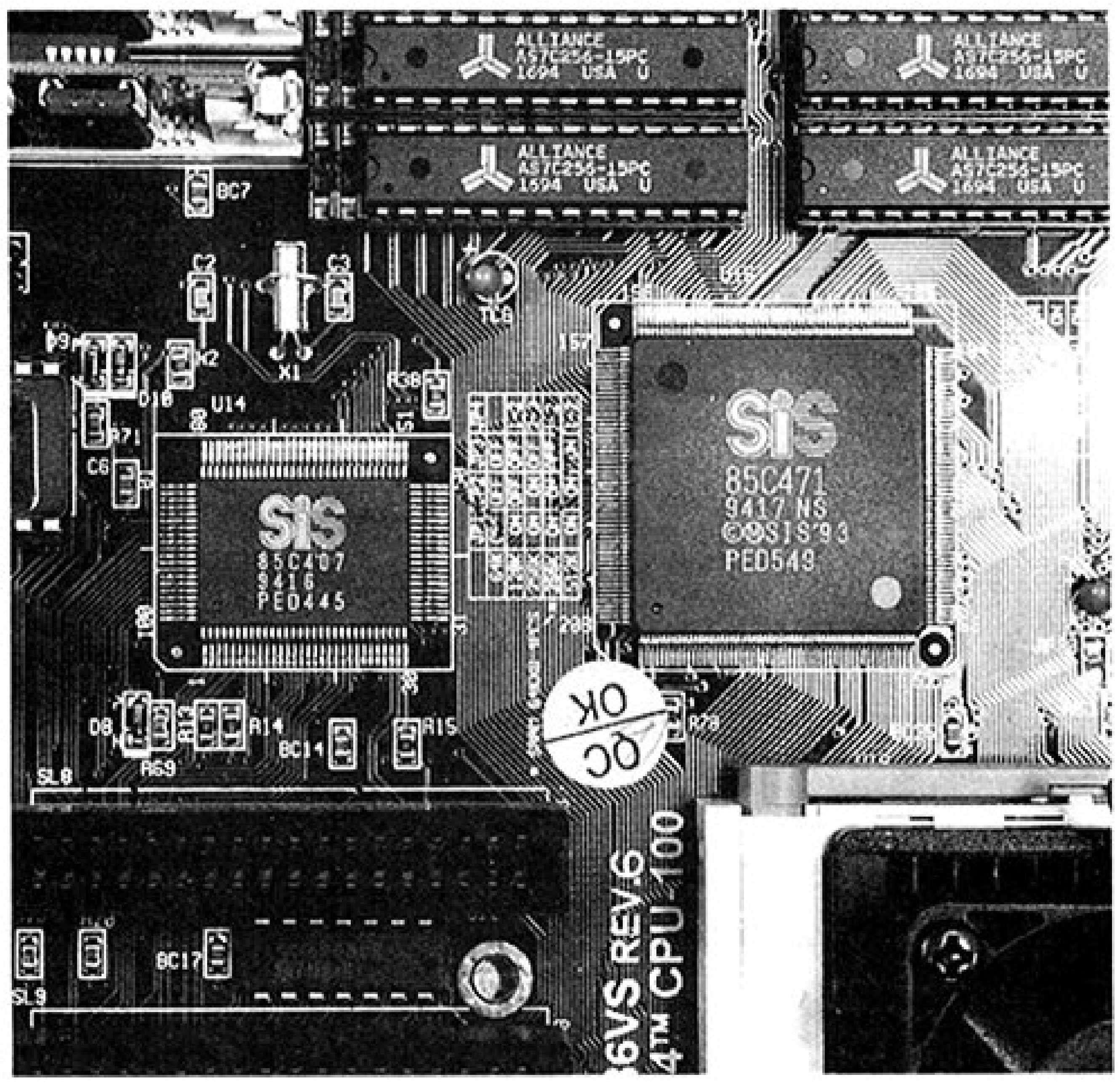
ただし，初期のVLバスはi486のバス信号線をそのまま使用していたため，技術的に最適な動作を行わせるには33 MHz動作が上限でした(他のCPUの場合にはブリッジデバイスを使用して，CPUのバスをVLバスに合わせる)．

これも徐々に改善され，40 MHzまでなら2枚，1枚だけなら50 MHzクロックでの動作が可能になります．

また，Pentiumの登場にともない，次世代のVLバス規格としてVersion 2.0も1993年に登場しています(筆者のPCはSCSIとSVGAがVLバス．お金がないのでPCIにはまだ買い換えられない)．



〈写真24〉 PCI バス付きマザーボード



〈写真25〉 チップセット(SiS)

PCI(Peripheral Component Interconnect)バス

1992 年 7 月に Intel が PCI バスを発表しました．最初に登場した PCI の規格は，PC 業界が求めていたローカルバスとはやや異なるものでした．

しかしながら，完璧なデザインルールに基づいた PCI の持つハイパフォーマンスと Intel が供給する PCI チップセットにより徐々に普及し始めました．

PCI は VL バスとは互換性のまったくないバスでしたが，1993 年に登場した PCI Release 2.0 は 64 ビットへの対応ならびに特定の CPU に依存しない仕様となり，従来の ISA，EISA，VL，MCA バスは PCI 上に拡張されたバスとして設計されています(**写真11，写真24**)．

そのため，PCI は従来のバスの置き換えを図ったバスでありながら，一般のマザーボードには PCI と ISA (あるいは VL も)のコネクタが付いているものがほとんどです．

チップセットとは

初代 PC が登場した頃は，CPU とメモリと周辺 LSI 以外はほとんどが TTL で，ボードには数十個の TTL がはんだ付けされていました．

初期のチップセットは，それらの外付け回路をまとめて一つのデバイスにして低コスト化と高信頼性を図った LSI として登場しました．

初めは，割り込みコントローラと DMA，その周辺の TTL デバイスを一つにまとめたものでしたが，ここ数年間に発表されたチップセットは，フロッピディスク・コントローラやシリアルデバイス，タイマーなどをまとめたものや，各 CPU のバスの仕様に最適化されたメモリコントローラやキャッシュコントローラを内蔵したものと，とにかく PC の用途と CPU の種類に応じて各社各様のチップセットが発売されています(**写真25**)．

特に，最近の PC 互換機のマザーボードに実装されている LSI は，表面実装されている 2，3 個のチップセットと，あとは周波数が選べるクロックジェネレータのデバイスぐらいです．

ちなみに PC のリアルタイムクロックは，もともと Motorola 製のものが使用されていました．Motorola も PC チップセットを作って売っているメーカーの一つです．

〈原文〉

巨蒼昔明秘
異徒既披靡
誕生互換機
昧允為永昌

〈読み下し〉

巨蒼昔、秘を明らかにし
異徒、既に披靡す
互換機を誕生せしむ
昧・允ために永昌す

〈大意〉

昔、青い巨人がその秘密を明らかにした
そのおかげで、違うアーキテクチャは
端に追いやられた
だが、互換機が誕生したおかげで、
結果的に繁栄したのは
マイクロソフトとインテルだった。

コラム●ドン・エストリッジという人物について

● PC 業界の人間ドラマ

PC 業界の歴史の中にはいろいろなドラマがあり，その中で色々な人物や会社が成功したり，失敗して消えていきました．そのようなドキュメントを綴っている書籍もかなり出まわっています．

そして，その中でも第一に挙げられるのが Microsoft のビル・ゲイツでしょう．

しかしながら，筆者の人生観から見た場合には成功した人物よりも，タイミングを逃したために消えていった Digital Research のゲーリー・キルドールや，強烈な個性と不屈の精神力で有名な Windows NT のデビッド・カトラーなどに興味がそそられます．

どの世界，どの時代でもその時に世の中が誰かを必要としているときに最適の人物が現れて一つの時代を作り，その時代とともに消えていくというドラマは，人類の歴史が続く限り永久に繰り返されるのかもしれません．

この世に生を受けたからには，そのような人生を送ってみたいと思う読者も多いことでしょうが，本人にとってはそういう運命のもとに生まれてきたために，結構たいへんだったり，他人に代わって欲しいのだけど他に代わる人がいなかったり，結構波乱万丈なんですよね(勝手に波乱万丈している筆者には片腹痛いことばかりです)．

ただ，この世の生命はそれぞれとつながりをもち，それぞれの役目を果たすために生きているというのは真理ですし，その役目が何なのか悟った上で(そうでなくても)，それに邁進できる人とそうでない人の差というものを最近ひしひしと感じます(米国に住んでいながら，手塚治虫のブッダを寝る前に読んでいるせいかもしれません)．

昔から「努力にまさる天才なし」といわれますが，強運をつかんだ「できる奴」が不朽の努力をしたから，みんながついていって一つの新しい時代が生まれるというのが本当のところのようです．

でも，そう言ってしまうと我々凡人の立つ瀬がないので，我々も努力だけはしようねっていう諺として生き残っているんでしょう．

● ドン・エストリッジの人生

さて，ここではそのような人物の中でも筆者の興味が強く揺さぶられた IBM のドン・エストリッジという人物について触れておきたいと思います．

彼はビル・ゲイツのような天才ではありませんでしたが屈強な IBM 社員であり，早くからパソコンに習熟していた人物だったそうです．

そして，IBM 本社からの束縛をあまり受けない形でスタートした PC 開発プロジェクトである「チェス計画」の責任者に抜擢され，IBM PC を世に送り出すことに全力を注ぎます．

彼の元には優秀なスタッフが多数集り，紆余曲折を経ながらも初代 PC は大成功を納め，ドン・エストリッジも一躍時の人になったのですが，どうあがいても彼らは IBM の社員であり，パソコン業界の他の会社は IBM PC の成功のおかげで巨万の富を築いているのに，彼らは自分達の成し遂げた業績に対して正しく報われることはなく，次々と優秀な部下が彼のもとを去っていきます．

また，中央集権的な IBM の首脳陣によって次々と彼の手足ももぎ取られていき，ボカラトンの IBM で始まった自由な雰囲気の PC プロジェクトも規模が大きくなるにつれて単なる IBM の一事業部となってしまいます．

最終的にドン・エストリッジ自身は昇格し，IBM 本社に戻されるのですが，プロジェクトの第一線から離れて，世界中に散らばる IBM の製造部門を管理する仕事につきます．

その直後，休暇をとるために乗っていた飛行機がダラス・フォートワース空港に向かう途中で墜落事故を起こし，彼は妻とともにこの世を去ってしまいました．

興味のある方は原本を熟読されることをお薦めしますが，IBM PC プロジェクトの成功は IBM でありながら中央集権的な束縛を受けないドン・エストリッジのような人物が責任者だったからこそできたのであり，彼もその役目が終わった直後にこの世を去ってしまいます．

まるで明治維新の前後に彼の能力が必要なときだけ活躍し，その後暗殺されてしまった大村益次郎(村田ぞう六，司馬遼太郎の「花神」の主人公)みたいな人だなぁと感じ入ってしまいます(ちょっとマイナーな人でしたね．大久保利通の方がピッタリかな)．

第3章

独自の道を進むPC98シリーズの過去と将来を探る

日本の国民機PC98シリーズの変遷

柳川誠介/WOK

PC98シリーズの歴史

● プロローグ

PC98シリーズは，最近でこそDOS/V機の登場でかつてのガリバー的な市場の独占はなくなりましたが，依然日本でもっとも使われているパソコンであることには変わりありません．

1982年に初代(無印の)PC9801が登場してからすでに13年が経過し，その進歩をたどることは半導体やコンピュータ周辺装置の技術の進歩をたどることにもなります．

技術の世界では，もっとも売れた製品が技術的にも優れているとは言い切れません．

しかし，技術の流れを引っ張ってきたのは，やはり「国民機」とさえいわれたPC98シリーズなのです．本稿ではPC98シリーズの歴史を振り返りながら，コンピュータ技術の進歩とこれからの流れについて考えてみたいと思います．

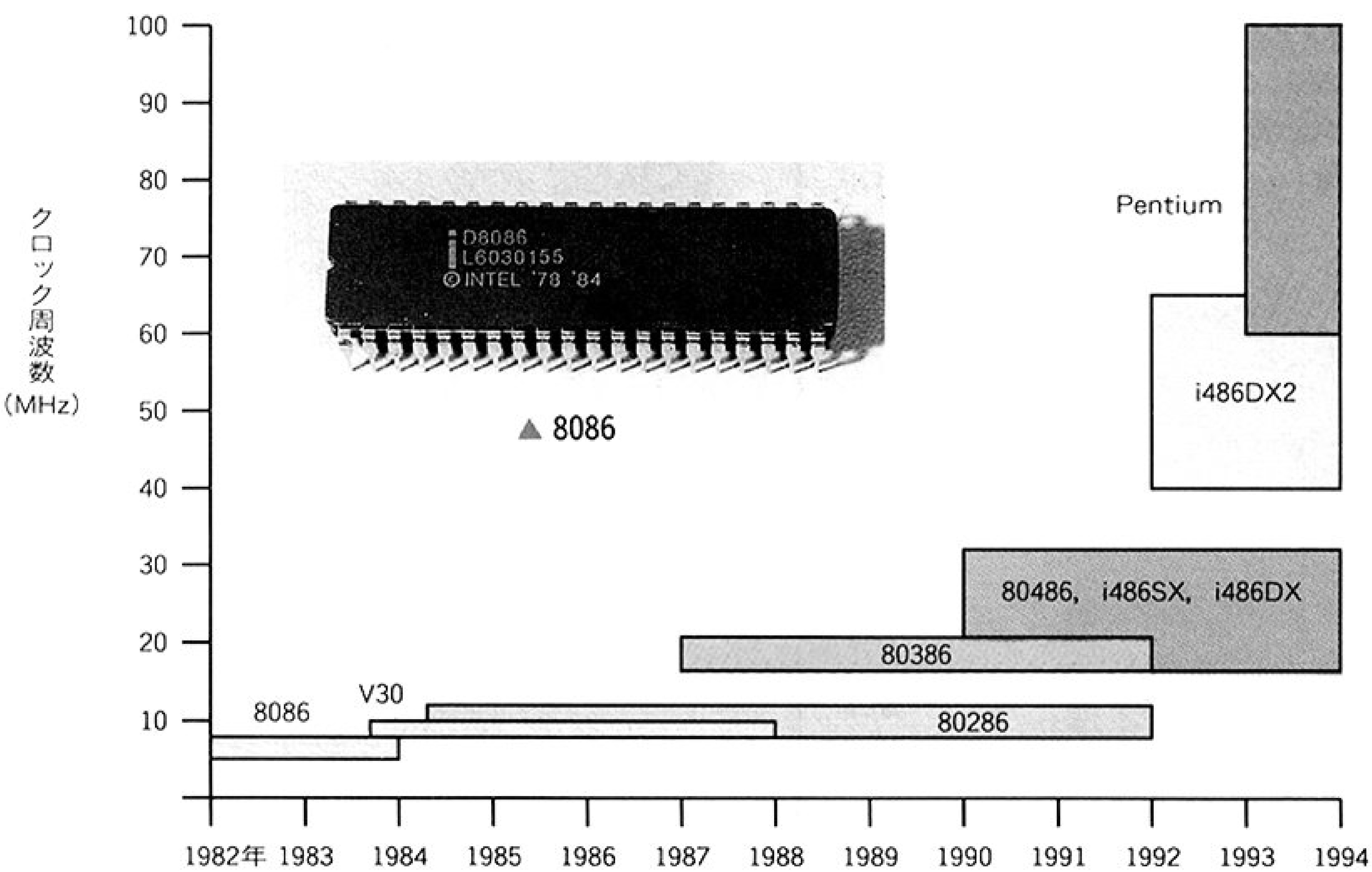
CPUとクロック周波数の変遷

図1がこれまでPC98シリーズに用いられたCPUの種類とそのクロック周波数を示したものです．クロ



〈写真1〉 PC9801E

〈図1〉
PC9801シリーズにおけるCPUとクロック周波数



ック周波数は20倍です．同じ周波数でもCPUの進歩により処理能力が上がりますから，実際の処理能力はそれをさらに上回ります．

筆者のところに現役で動いている1983年製のPC9801E(**写真1**)が出た頃，当時ベンチマーク・テストでよく用いられた素数を求めるプログラム**sieve**(**リスト1**)は，クロック周波数を5MHzにしたとき実行に38秒かかりました．

同じプログラムをi486DX2の66MHz機で実行すると0.47秒ですみます．現在のPentium 90MHz機では処理能力は初代9801の100倍を越しているでしょう．

工業製品の歴史のなかで，これほどまで短期間で性能の向上がなされたものがほかにあるでしょうか．考えてみればたいへんなことです．

〈リスト1〉ループの処理スピードを評価するために用いられた素数を求めるプログラム(sieve)

```
#include    <stdio.h>
#define SIZE    8192
#define SIZEP1  8191
#define TRUE    1
#define FALSE   0

char    flags[SIZEP1];

main()
{
    int     k,i,prime,count,iter;

    printf("100 iterations¥n");
    for(iter = 0; iter < 100; ++iter){
        count = 0;
        for(i = 0; i < SIZEP1; ++i)
            flags[i] = TRUE;
        for(i = 0; i < SIZEP1; ++i){
            if(flags[i]){
                prime = i + i + 3;
                for(k = i + prime; k < SIZEP1; k += prime){
                    flags[k] = FALSE;
                    ++count;
                }
            }
        }
    }
    printf("¥007%d primes¥n",count);
}
```

メモリ容量の増加

CPUの処理能力が上がれば，豊富な機能を持ったアプリケーションソフトが楽に実行できるようになります．

それとともに，そのようなアプリケーションソフトは多くのメモリを必要とするので，実装されるメモリ容量が増加します．**図2**は，標準装備されるRAMの容量の変遷です．

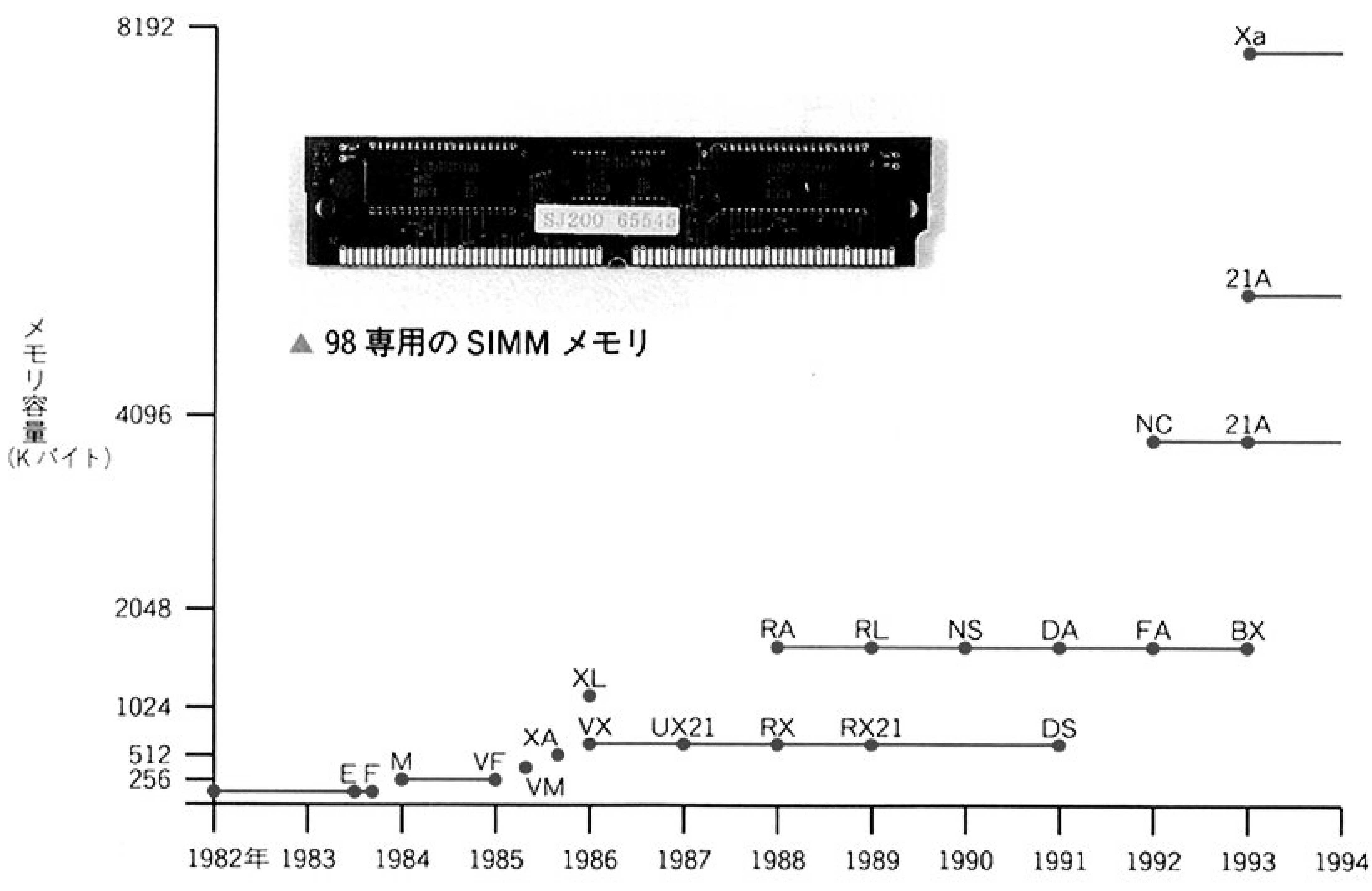
パソコンを含むコンピュータの応用領域はCPUの処理速度の向上だけでなく，高機能なプログラムの開発と低価格な大容量メモリの普及がともなってこそ，広がっていくのです．

図2を見ると，640Kバイトおよび1372Kバイトの期間が長いことに気がつきます．ハードウェアの面からは，CPUに80286が採用されてから以降は，1Mバイト以上のメモリ空間も扱えるようになりましたが，メモリを増設してもRAMディスク程度の用途がほとんどでした．

その理由はMS-DOSを使用している限り，640Kバイト以上のメモリを有効に活用できなかったからです．

本格的にメモリが活用されるようになるのは，i486の時代に入ってからで，Windowsの普及を待たなければなりませんでした．

〈図2〉PC9801シリーズの標準装備のメモリ容量



▲ 98専用のSIMMメモリ


▲ Cバス用メモリボード

互換性を保ちながらの発展

初代98が出た後，後継機種として大幅に構成を変えたものを出そうという企画がなくはなかったそうですが，互換性を重視することが優先されました．

そうして生まれたのがPC9801E/Fで，それ以降CPUはいろいろ変わってもハードウェアの基本は「PC98シリーズのソフトはどの98でも動く」という基本に変化はありません．これがあったからこそ，多くのユーザーの支持が得られ，圧倒的な市場占有率を得るに至ったわけです．

そもそも，86系のCPUの発展過程こそが互換性を保ちながらいかに性能を上げるかに苦労してきた歴史です．

8086はその前身の8ビットCPUの8080, 8085の発展形であり，186, 286, 386, 486そしてPentium CPUまで，すべて過去の遺産を背負って設計されています．

これは世界のパソコンの70%以上に86系CPUが使われ，そのほとんどがMS-DOSを搭載するに至った現状に縛られた結果ともいえなくもありません．

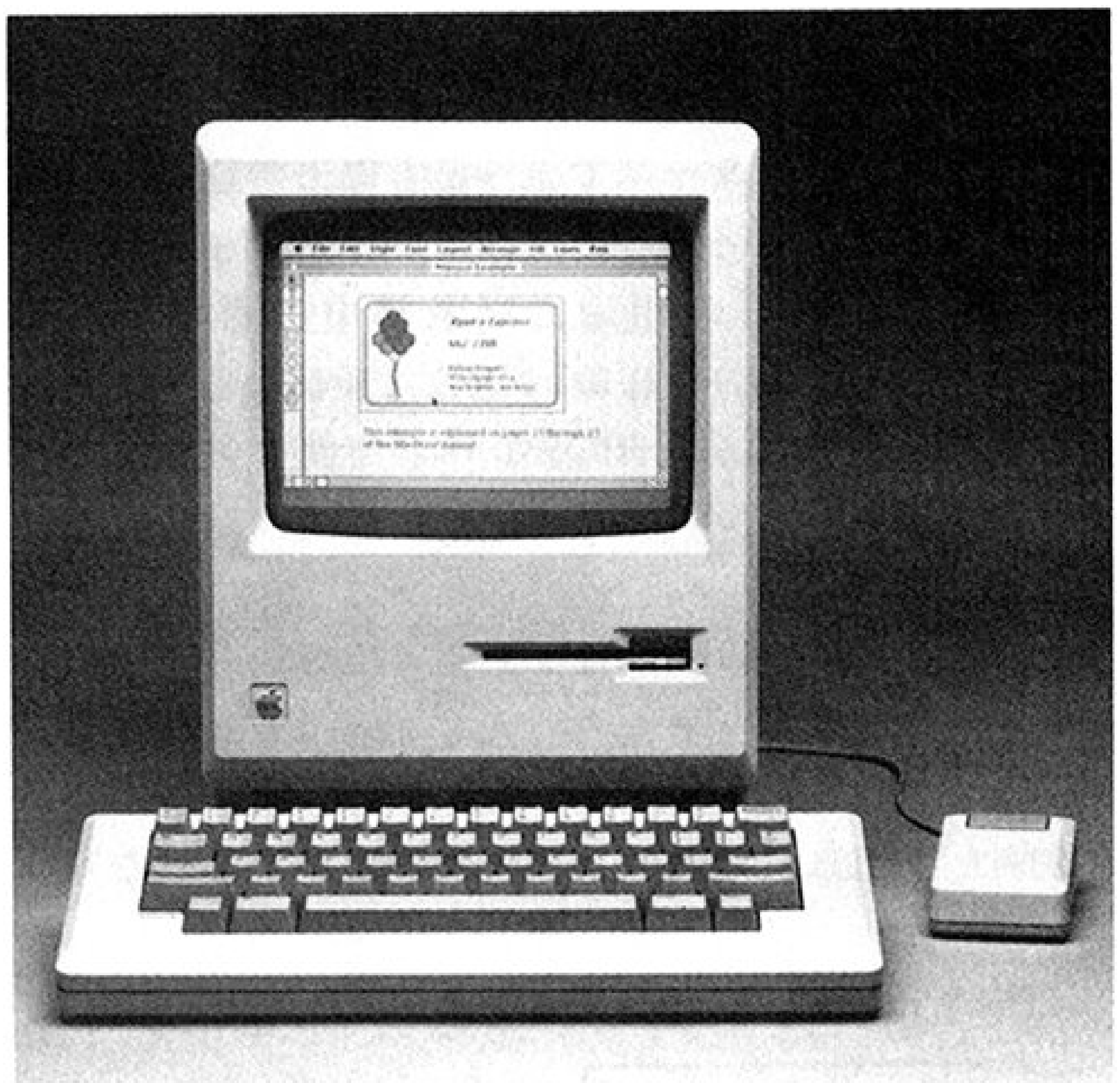
どんなに高性能なCPUを開発しても，それまでのソフトウェアの資産が生きなければユーザーに受け入れられないからです．

しかし，技術的な側面から見ると，互換性を保つことにこだわり，過去の設計思想を残すことは進歩を阻害する要因にもなります．

PC98シリーズの歴史を振り返ってみると，互換性と性能向上の狭間で苦労した跡がいくつか見られます．

先発ゆえの苦悩

16ビットCPUで最も先に普及したのが86系であり，それがIBMのパソコン(1981年発売)とMS-DOSに支えられ，市場の大半を制してしまったことはあえて解説するまでもありませんが，進歩の速い技術の世界，後を追うパソコンメーカーはその上を行こうと努力してきました．



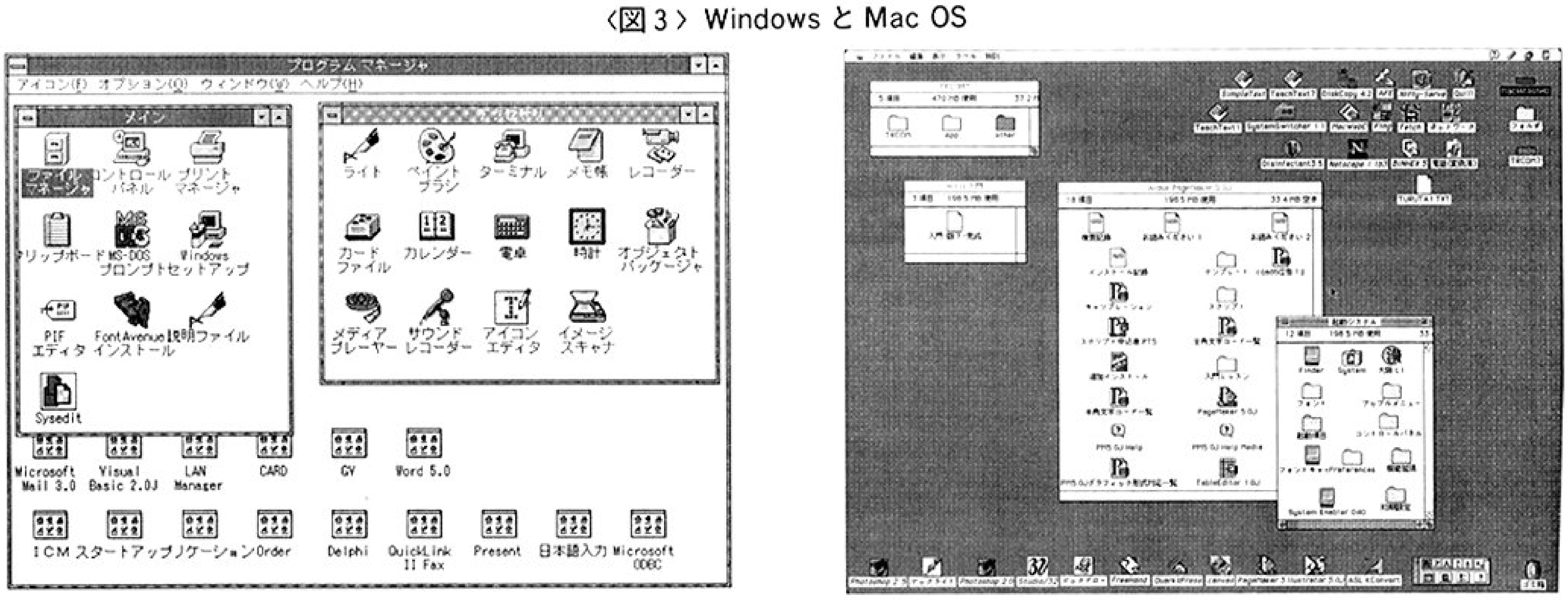
〈写真2〉初代Macintosh 128

Intelの86系CPUに対して，何かと比較に出されるのがMotorolaの68系MPUで，そのMPUを搭載したパソコンといえばAppleのMacintoshです(**写真2**)．

初代Macintoshの発売は1984年ですが，68000 MPU自体がリニアなアドレス空間を持っていたため，80386が出る以前に多量のデータを使う印刷やデザインの分野で支持を得ました．

また，そのグラフィカル・ユーザーインターフェース(GUI)が今日のWindowsに大きな影響を与えたことはいうまでもないでしょう(**図3**)．今日でも，価格は別として，標準装備されているインターフェースや拡

〈図3〉Windows と Mac OS



(a) Windows　　(b) Mac OS

張ボードのPlug and Playの考え方，などに出発時の3年の差が残っているように思います．

現在のPC98シリーズでもっとも過去の遺産に引きずられているところはシリアルインターフェースでしょうか．LSIにこの用途では老舗中の老舗である8251A(その互換品)を搭載，NECは通信機器のメーカーでもあったためか，BIOSレベルで割り込みをサポートするなど，初めのうちはIBM機より豊富な機能がありました．

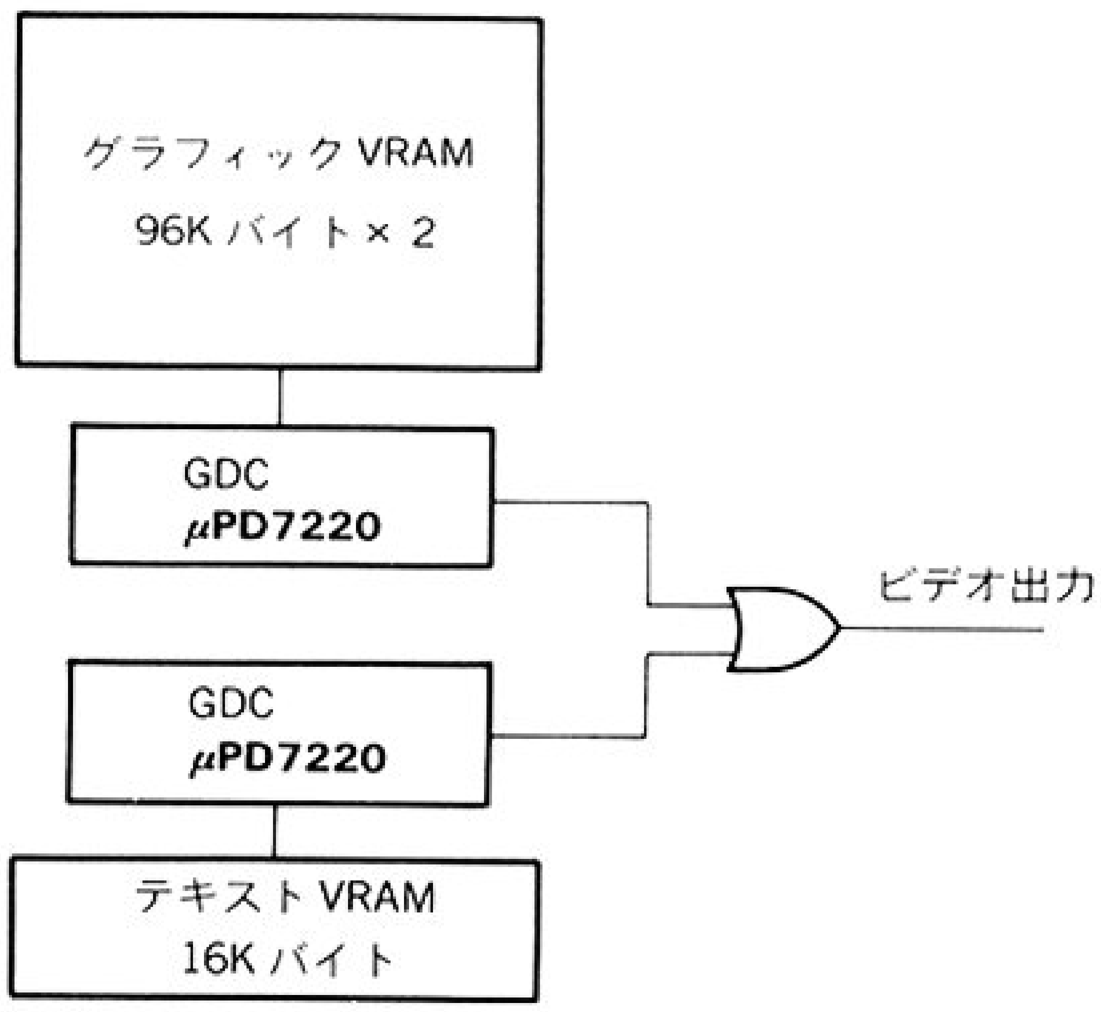
しかし，その後IBM機はシリアルポートを拡張し，4チャネルをサポートしたうえに最高19200 bpsまで通信できるようになりました．PC98シリーズでは9821Af，9801BA2になって19200 bpsが可能になりましたが，標準のチャネル数はまだ1です．ようやく，今年の5月15日に発売された最新のXaシリーズにおいて，115 Kbps×2チャネルに変更されました(このうちの1チャネルはIBM PCと互換)．

速度に関してはインターフェースLSIの能力で上限が押さえられます．最近のIBM機に搭載されているNS16550Aは256 Kbpsまでの能力があり，インターフェースにRS-485などを用いれば，相当高度な通信ができます．

現在のPC98シリーズに搭載されているLSIは本舗8251Aよりは進歩したものなっていますが，根本的に考え直す時期にきたと考えられます．

なお，Macintoshでは，はじめからRS-422を採用し，230.4 Kbpsまで使えます．しかも，RS-232Cと互換性のある方法をとっているのが憎いところです．

〈図4〉PC9801シリーズのビデオシステムの基本型





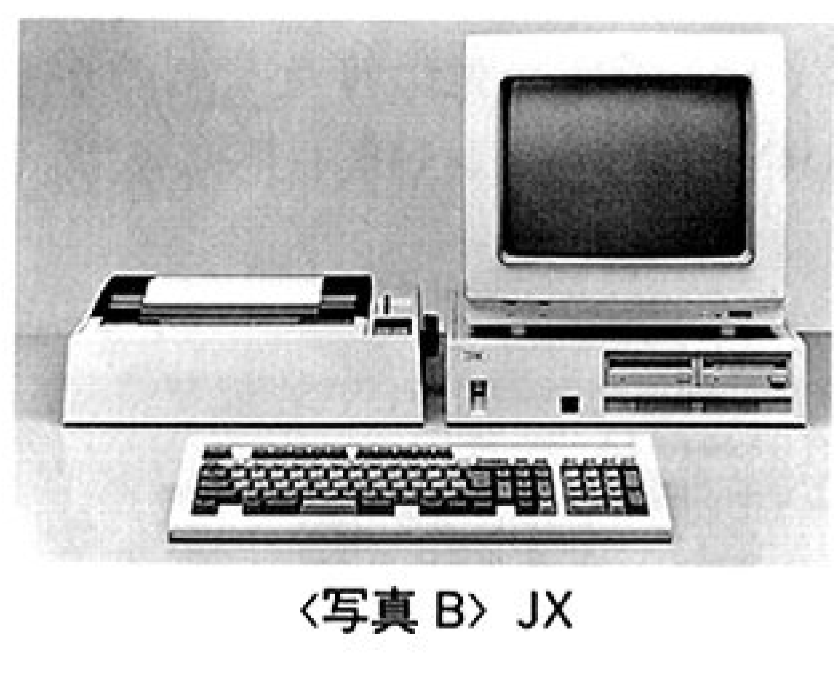
〈写真3〉PC98XL(XAの後継機)

コラム● DOS/V以前の日本アイ・ビー・エム製パソコン

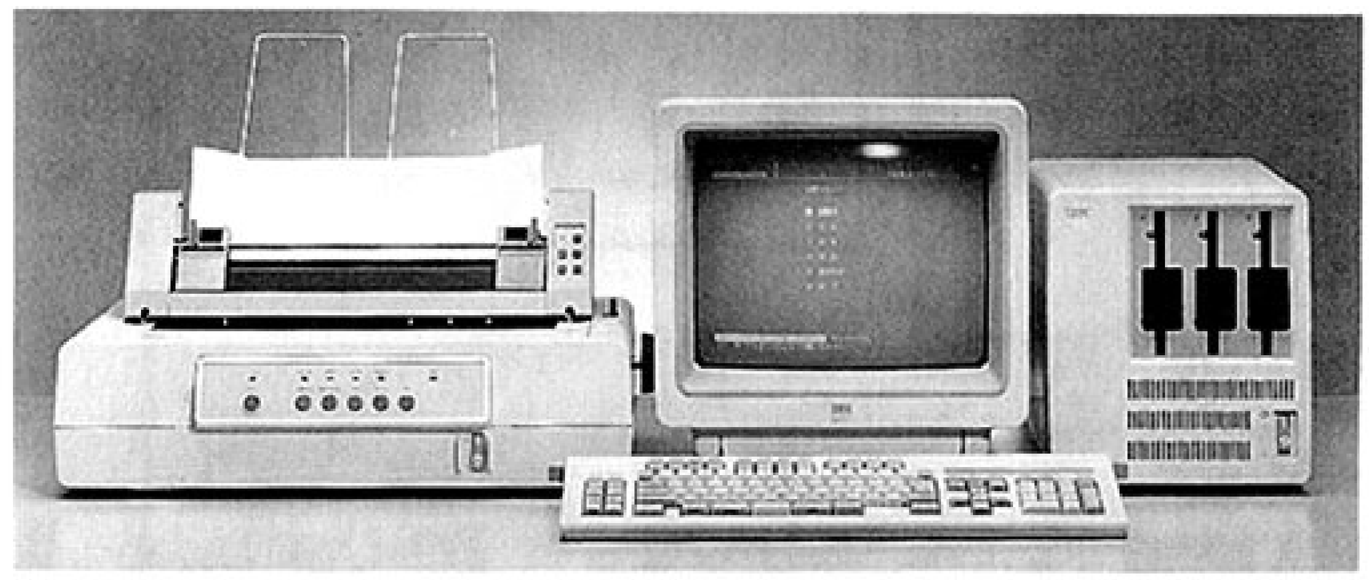
DOS/V以前から日本アイ・ビー・エムもパソコンを販売していました．その一部を**写真A～写真C**に示します．



〈写真A〉PS/55



〈写真B〉JX



〈写真C〉5550

運の悪い選択をした？ ハイレゾモード

PC98シリーズは初めから640×400ドットと，IBMのパソコンと比べると，高解像度の画面を持っていました．初期のIBM機は640×200ドットで，PC98に追いつくのはCADなどで高度なグラフィック処理をやるようになってからです．

PC9801は初代のモデルからグラフィック処理では当時の最高水準にあったμPD7220(GDC)を2個搭載，一方をキャラクタに，もう一方をグラフィックに使用，両者の和を映像信号として得ていました(**図4**)．BASICの描画などは当時の最高水準の速度であったことでしょう．

やがてソフトウェアの高度化に伴い，より高度な解像度が要求されてきます．PC98シリーズでは1985年に1120×750ドットのXAが登場します(**写真3**)．

以後この解像度(ハイレゾリューション．通称ハイレゾ)を持ったシリーズは，PCH98シリーズとしてパソコンのハイエンドに位置付けられる方針でしたが，機種も，出荷台数も微々たるものにすぎず，Windowsの時代到来と共に事実上姿を消します．

その原因は互換性のなさでしたが，それを乗り越えるソフト側の努力も足らなかったと思います．現在，ノーマルの解像度しかないPC98にはアダプタボードを付加してハイレゾのディスプレイを行う方法がありますが，この手法をもっと早くから取り入れるべきでした．

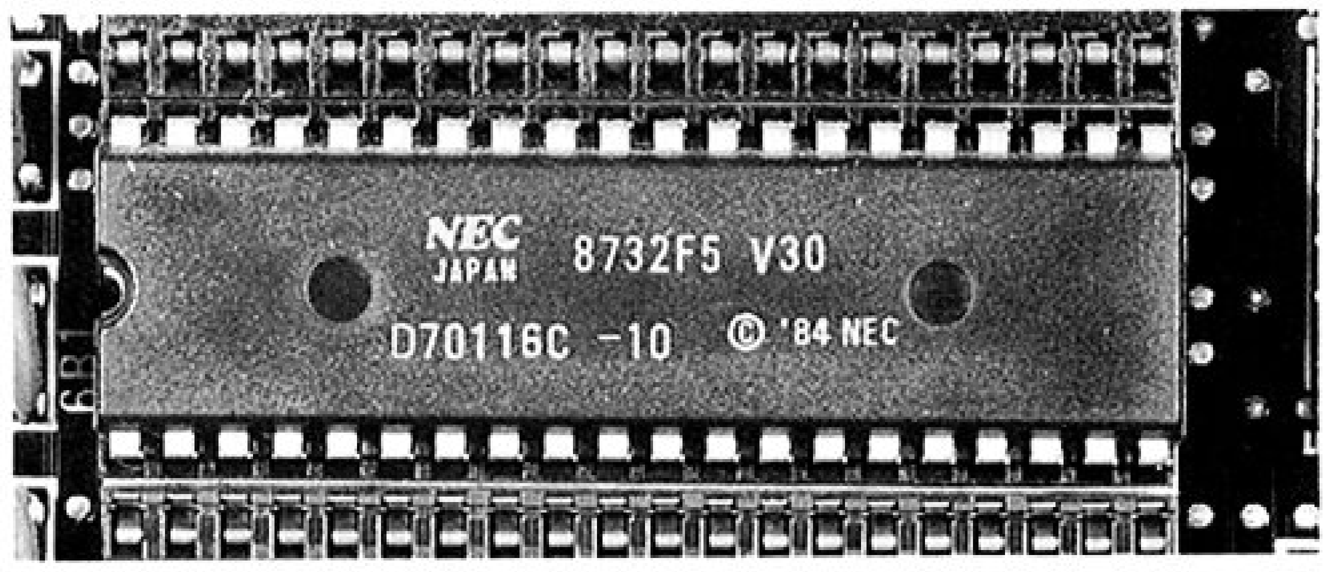
当初解像度が低かったIBMのパソコンでは，その後何回も解像度のアップを行っています．注目すべきは，マシンのすべてを交換しなくても，拡張スロット上のビデオアダプタのみを交換すればグレードアップできることです．

ソフトウェアの側では，装着されているアダプタがどのタイプかチェックし，ビデオドライバをセットアップします．

多くのビデオアダプタは複数種のサイズのCRTをサポートし，CRTのサイズが大きくなればそれに従って表示範囲も広がるようになっています．

〈表1〉よく用いられるビデオモード

ドット数	カラー	名　称	CRTサイズ(インチ)
512×384	256	Macintosh	12
640×480	16	VGAモード12H	14
640×480	256	SVGAモード101H	
640×870	グレー256	Macintosh	15(縦長)
800×600	32768	SVGAモード113H	17
832×624	32768	Macintosh	
1024×768	32768	SVGAモード116H	21
1152×870	256	Macintosh	
1280×1024	32768	SVGAモード119H	



〈写真4〉V30 CPU

このように柔軟なビデオシステムを取り入れたことにより，IBM PCの世界では多くのサードパーティがビデオアダプタ市場に参入し，それぞれが互換性を持ちながら性能を競っています．14インチの場合，640×480ドットが世界標準で，PC98シリーズも1992年のPC9821シリーズからこの表示が切り替えによって可能になりました．

このように表示能力で遅れをとったのは，漢字を表示する必要から相対的に高解像度で出発したPC98シリーズの「恵まれていた故の油断」だったかもしれません．

残念に思うのは，ハイレゾモードのドット数が1120×750でなく，1024×768とか1152×870だったら，IBM，またはMacintoshのどれかと互換性があり，そこから国際的な規格へ結びついていったかもしれないことです(**表1**)．

Vシリーズは互換CPUビジネスのはしり？

1985年発売のVF，VM，UではCPUは8086ではなく，μPD70116(V 30)が搭載されています．V30は8086に差し替え可能で同一クロックで2割程度高速であり，独自の命令も持っています．

その後，V30のみを搭載した機種はなくなっていきましたが，i486が主流になる1991年頃まで一部機種にはIntelのCPUとともに搭載されていました．筆者も8086をV30に差し替えて使用したものです(**写真4**)．

いまはCPUの差し替えなど当たり前で，AMDやCyrixが本家Intelとつばぜり合いをしていますが，パソコンの性能アップといったら買い換えるしかなかった当時はV30の登場は画期的なことでした．

しかし，NECとIntelの間で著作権問題が起き，それが決着するためにかなりの時間がかかり，結局日本のメーカーは80286以降の互換チップ市場から引き下がるということになります．

競争力のあるCPUを開発する力はあったのに，CPU市場のほとんどを米国メーカーに押さえられている現状は残念です．

2転3転した拡張スロット

32ビットのCPU 80386が登場すると，それまで98バスと呼ばれていたバスでは性能的にCPUの足を引っ張る形になりました．

そこで1991年，98バスの隣に2階建てになるような形で32ビットCPU関係の信号を出すことにしました．これはNESAバスと呼ばれ，ハイレゾ機種を中心に装備されました．

しかし，NESAバスを使った拡張ボード自体があまり開発されず，現在このバスを使った機種は生産されていません．

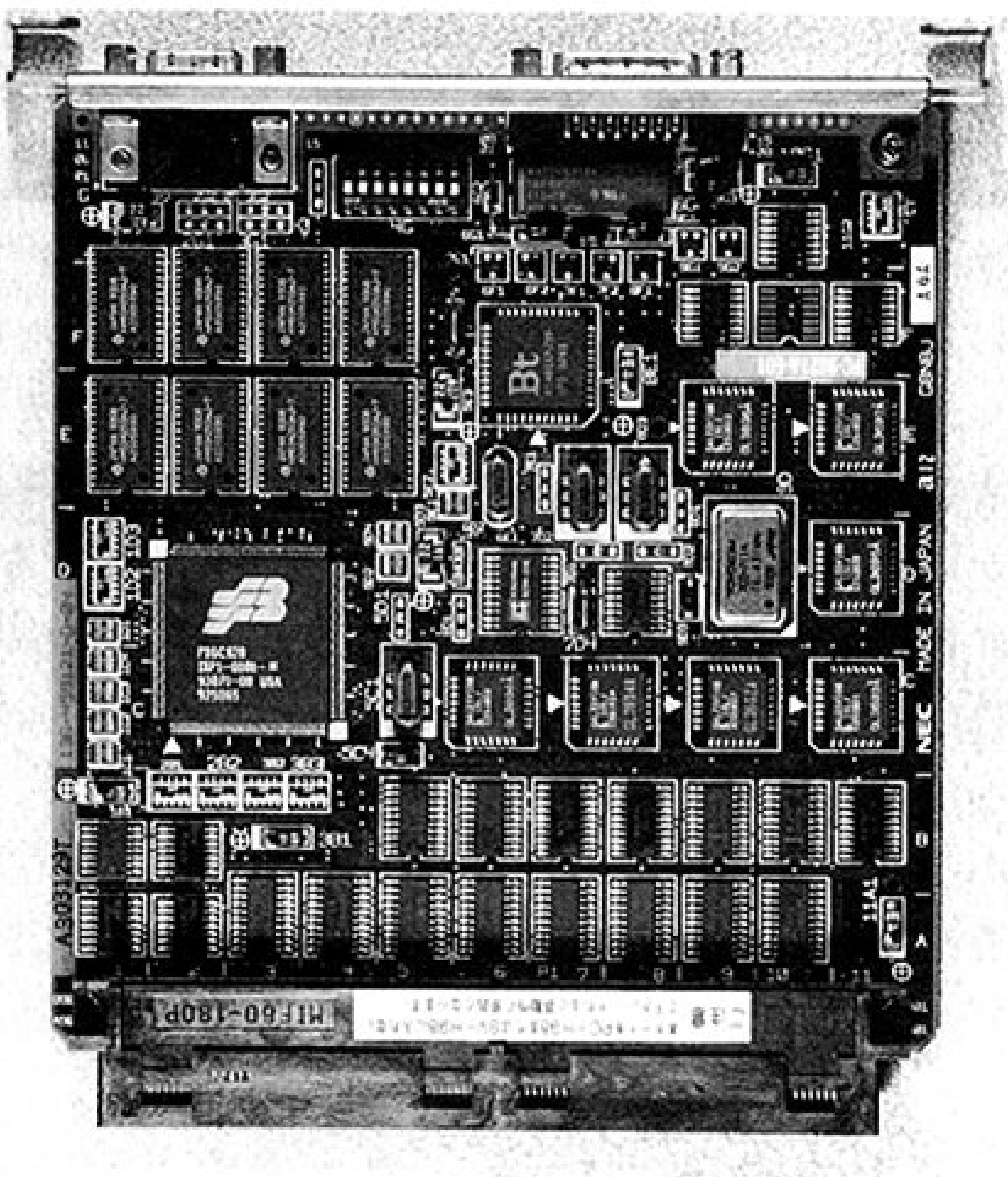
次に登場したのがファイルスロットで，周辺装置が取り付けられやすいようにSCSIなどのインターフェースを前面に設けたものです．

これはその後，ファイルベイと呼ばれるスペースになり，そこにケーブルのみが用意されている形になりました．

この形態はすでにIBM互換機で行われている方法で，安価に出回っているIDEインターフェースのハードディスクなどの周辺装置が使いやすくなりました．

ファイルスロットとほぼ同じくして採用されたのがローカルバス・スロットで，これはCPUの能力をより発揮するために設けられた高速バスです．

これはIBM互換機のVLバスに近いといわれていましたが，技術情報の公開もなく，対応製品もほとんどないままPC9801BシリーズとPC9821Aシリーズ



〈写真5〉PC9821Asのローカルバス用ボード



〈写真6〉JSWORD

の一部に採用されたままになっています(**写真5**).

このように，バスに関しては，規格をオープンにして，多くのメーカーがそのバスに対応したボードが作れるようにしないと成功しないようです．

最新機種では，パソコン用バスの標準規格として定められたPCIバスが設けられています．今後これがどの機種にも搭載されていくことでしょう．

「一太郎」より進んでいた？「一太郎の前身」

一頃パソコンの「国民機」といえば98，そのワープロソフトといえば圧倒的に「一太郎」と答えが返ってきた頃(今でもそういえるでしょうが)，筆者は「一太郎」のユーザーインターフェースをよいものとは思っていませんでした．

ごく一般の人があのようなものを使うことにより「パソコンとはこんなもの」という印象を持つことは，コンピュータの普及の妨げになるとさえ思いました．

確かに，バージョン4までのメニューによる機能の選択方法は古いといわれてもしかたないのですが，一太郎が「国民的ワープロソフト」になる前に，ジャストシステムはもっと先進的な考え方をワープロソフトに取り入れていたようです．

1984年，まだジャストシステムの製品はアスキー経由で発売されていましたが，当時新発売のワープロソフトJS-WORDの広告を見ると，画面上部に今日のものにそっくりなアイコンを備え，マウスによるカット・アンド・ペースト，画面分割などができるとうたっています(**写真6**).

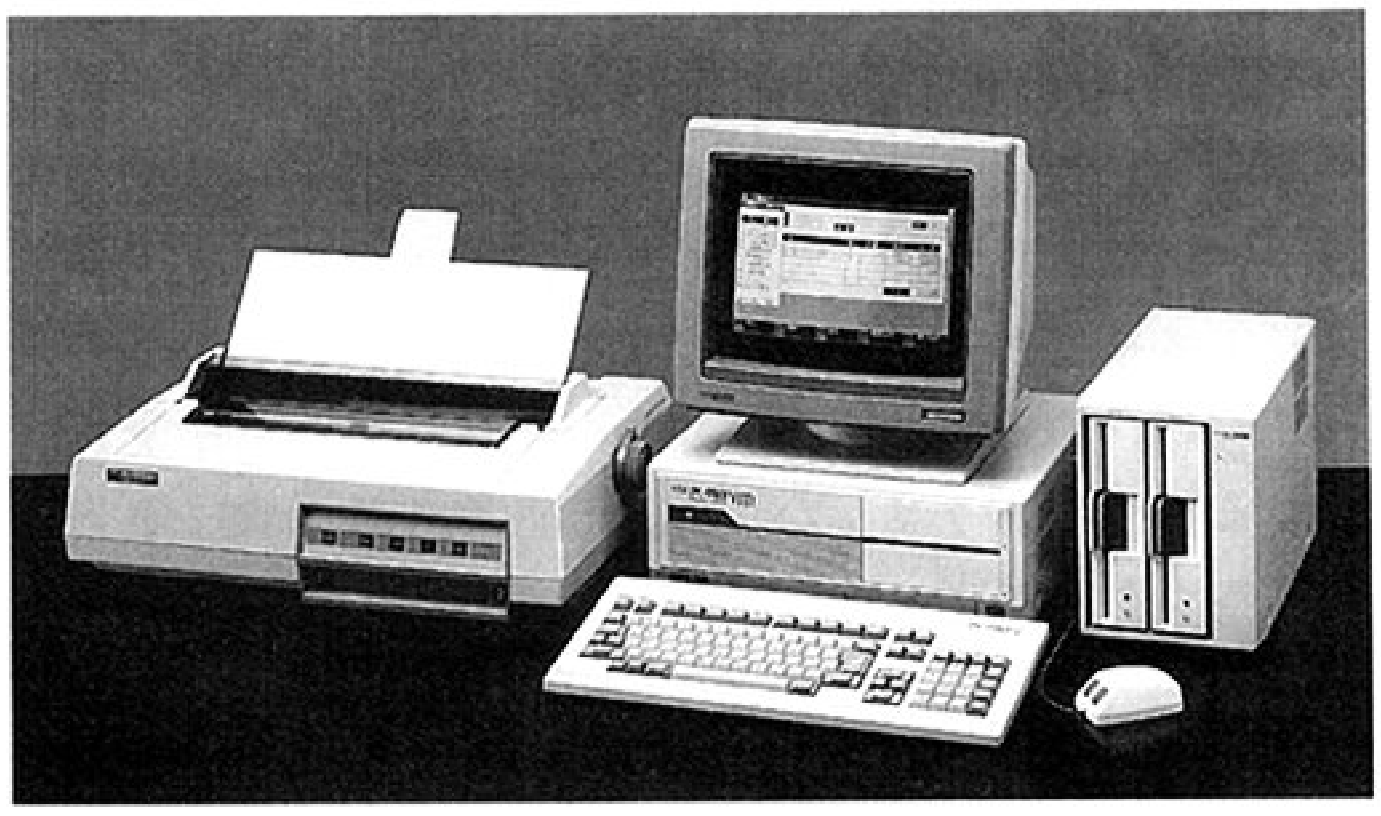
それでも人気が出なかったのは，価格が6万円で，マウスがない場合はさらに2万円というのが原因でしょうか．それとも，当時の日本のコンピュータ文化はそのようなユーザーインターフェースを受け入れるほど成長していなかったからでしょうか．いずれにせよ，MS-DOSの仕様にマウスがあったら今日のソフトの様相はかなり変わったことでしょう．

ちなみに，この原稿は98に関することながら，MacintoshにATOKをのせて書いています．

初期の98に必要だった「仕事始めの儀式」

現在のPC98シリーズに用いられているカレンダ時計はμPD4990Aまたは相当品ですが，無印PC9801からVM，UVそしてXAまではその前身のμPD1990Cが用いられていました．このIC，月日と曜日のレジスタはあるのですが，年のレジスタがありません．年号は**DATE**コマンドでセットしたものが保持されるだけです．

つまり，1994年12月31日の次は1994年1月1日．



〈写真7〉 PC9801Vm

したがって，これを知らずに新年早々ソースプログラムを手直しするとセーブしたファイルは皆古いものとみなされ，makeツールが何もしてくれないハメになります．

新年初出勤の際には「心を新たに」して**DATE**コマンドで年号をセットしたものです．

同様なことが年のレジスタの値が99から00に変わったとき，すなわち2000年1月の仕事始めのときに，あちこちの職場で行われるのではないかと思うのですが，5年後はどうなるでしょうか．

8237AでDRAMのリフレッシュをする

VM21(**写真7**)まではDMAコントローラ8237Aのチャネル1はDRAMのリフレッシュに用いていました．その後，メモリの高速化やリフレッシュモードの多様化で，リフレッシュはメモリの周辺回路で行うようになり，このチャネルはディスク用に使われています．

小規模な，さほど高速なDRAMを使わないシステムではDMAコントローラによるリフレッシュは簡便でよいかもしれません．

それにしても，CPUはi386になって64Kバイトの「セグメントの壁」が取り払われたのに，DMAコントローラは旧態依然です．

NECでも，高機能なDMAコントローラLSIを開発していますが，8237Aと互換性がないためか普及しません．一方でCPUが高速化するにしたがって，DMAを用いなくてもかなりのことができるようになっています．

PC98におけるDMAはそろそろ過去の技術になりつつあるのでしょうか．

互換機の登場とDOS/V機の影響

PC98シリーズの互換機をエプソンが出したのは1987年4月ですが，当時はその他にも互換機にまつわ

るいろいろな動きがありました．同じ頃，シャープがMZ-2861というPC98のBIOSエミュレーション機能を持つパソコンを発表しましたし，一方ではAX協議会が組織され，PC/ATの日本語版の仕様をまとめ，三洋電機などが製品を発表しました．

さらに，PC9801とPC/ATの双方のソフトが実行できる機種もありました．このうち互換性の完全さと性能の点からエプソンだけが残ったわけです．

漢字表示ができるPC/ATの互換機といってもDOS/Vではありません．EGAと呼ばれる640×350ドットのビデオアダプタに，漢字表示機能をつけ加えたJEGAなる日本語用のビデオアダプタを差し替え，640×480ドットの表示を行うものです．漢字のフォントはROMで持ちます．このような非英語圏への対応方法は日本以外でも行われていましたが，ハードウェアの変更を伴うため，価格と互換性の問題で普及せず，そのことがDOS/Vの開発を促します．

DOS/Vにおける漢字表示はフォントをソフトウェアによりグラフィック表示するというものです．MS-DOSのもとで多様なフォントを用いるソフトウェアはアマチュアも含め，いくつかのところで開発されていましたが，1990年に日本IBMが正式に仕様を統一してから急速に普及しました．

これによって安価なIBM互換機がどっと入ってくる，まさに「黒船の到来」と騒がれたものです．パソコン界初めて？の比較広告も登場しました．

しかし，PC98用のMS-DOSのプログラムでDOS/V用に書き直されたものはそう多くはなく，今となっては，あの騒ぎはWindowsへの前奏曲にすぎなかったのだと思います．最大の功績はDOS/VがきっかけでPC98シリーズが安価になったことでしょうか．

以後，PC98シリーズはモデルチェンジするごとにIBM互換機との互換性を高めていきます．

21世紀のPC98シリーズは

かつてPC98シリーズにつけられた「国民機」という呼び名には何となく閉鎖的なイメージがあります．今，Windowsをはじめとする新しいソフトウェア環境が整うにつれ，日本独自の仕様を持つことはもはや意味がなくなりました．

マルチメディアの時代を迎え，これからのソフトウェアは世界のどの機種でも通用することが重要なポイントとなります．「国民機」の時代は終わったのです．

しかし，消えていった，あるいは消えていく運命にあるいろいろな工夫がすべて価値がないものだったのでしょうか．

それどころか，時代の最先端を行っていたものもいくつかありました．ただそれが世界の流れに今一つ乗り切れなかっただけだと思います．

その創意を今後も絶やすことなく製品に盛り込んでいただきたいと思います．これは筆者個人の願いであるのみならず，技術立国日本の生き残る道でもあるのではないでしょうか．

見方によっては，現在はパソコンのハードウェアの転換期ともいえます．まずはWindows 95から本格的に取り入れられる予定のPlug and Playの考え方です．

これによってパソコンユーザーは拡張ボードを装着する際のいろいろな設定から解放されます．Plug and Playでは同時にインターフェース，ディスプレイなどの仕様が新たに定められることになっており，これらにPC98シリーズがどう対応するか，DOS/V機の登場以上に注目されます．Plug and Playの思想がどこまで浸透するかはMicrosoftの主導権が鍵になります．

もう少し先を見れば，マルチOSの時代が予想されます．現在，86系のパソコンではWindows, Windows NT，OS/2がそれぞれ得手不得手の分野はあるものの，覇を競っています．

その一方では，IBMが，PowerPCを用いたOS/2を計画しています．これはCommon Hardware Reference Platformと呼ばれるハードウェア上で稼働され，そこではOS/2ばかりか，UNIX, MacintoshのOSも動作可能というのです．

こうなると，いかに多くのOSをサポートできるかがパソコン選択の鍵となって，CPUに何を使うかなどは二の次の問題になってきそうです．

したがって，86系，68系，CISC，RISCなどの派閥にこだわることはなくなってくるのかもしれません．これまで日本のメーカーは何回も独自CPUを開発しては主導権を取れずにきました(TRONチップもしかりです)が，今度こそ，性能で勝負できる状況になってきたといえないでしょうか．

PC98シリーズの歴史はパソコンの技術史であり，日本の技術思想の反映でもあるのです．NECをはじめとするメーカーにさらなる奮起を期待したいところです．

〈柳川誠介〉



〈写真8〉最新のPC98(98 MATE X)

Appendix

プログラミング言語はどのように進化したか
プログラミング言語変遷史

プログラミング言語の歴史は，人間が理解できることをコンピュータに理解させるという方法の進化の歴史ともいえるでしょう．別な見方をすれば，人間とコンピュータとの妥協の産物ともいえます．

ここでは，プログラミング言語の歴史の概略を解説します．

● 最初の言語

コンピュータのための最初の言語はマシンコードで，2進数(または16進数)で表されたものでした．今でもコンピュータの初歩を学ぶ人は，2進法について学ぶことが通例です．

2進法は，コンピュータのしくみを反映しているだけで，なんら人間にとって使いやすいものではありません．したがって，10進法の演算を行うための特別なテクニックと多大な労力を伴いました．

事実，日常を見渡してみると2進法を必要とする場面は皆無です．

● マクロ

コンピュータは，もともと高速で正確に計算をするために生まれてきたものなので，加算ルーチンのような基本演算は頻繁に使われていました．

頻繁に使われるものを何度も記述するのは煩わしいということで，次第に加算ルーチンはひとまとまりのマクロ(**add**)となりました．これはマイクロプロセッサにも搭載されるようになります．

● アセンブリ言語

マクロ化が進んでも，プログラミングが面倒であることにあまり変わりはありませんでした．例えば，68000マイクロプロセッサで，d0レジスタの内容16ビットを1000/1001番地に移動するには，

```
0011 0001 1100 0000 0001 0000 0000
0000   (=31c01000)
```

と書きます．

しかし，このような数字の並びが100も500もあれば，どんなに忍耐強い人でも耐えきれないので，よりわかりやすい形で記述する方法が考案されました．

それがアセンブリ言語で，上記プログラムは，

```
move.w d0, $1000
```

と記述します．

少しは，わかりやすくなったといえるでしょう．

● コンパイラ

アセンブリ言語では，コンピュータが異なると一から言語の学習をし直す必要がありました．

また，少しは人間にとってわかりやすい記述ができるとは言え，コンピュータをじかに操作しているため，コーディング効率が悪く，バグが入りやすくなります．そこで，FortranやCobol，Algolなどのコンパイラ言語が生まれました．

Fortranは，ご存知のとおり今でも使われている科学技術計算に適したコンパイラ言語です．

Cobolは，英語のわかる人なら理解できるように作られたと言われる言語で，主に事務処理に使われています．

Algolは，もともとコンピュータ言語ではなく数学の世界で試みられた定式化の手法でした．たまたま「Cマガジン(ソフトバンク刊)」の1995年5月号に詳しい解説がありますので，そちらを参考にしてください．

Algolは，言語としては成功しませんでしたが，再帰(自分自身を呼び出すルーチン)を生み出すなど，理論上の貢献は大きいといえます．

● BASIC

コンパイラ言語の次に生まれたのが，BASICインタプリタでした．

BASICは，覚えるのが簡単で規模が小さく，何より即実行が可能なインタプリタだったので，瞬く間に普及しました．

8ビットパソコンから初期の16ビットパソコンには必ずといってよいほどBASICが搭載されており，パソコン創世紀の頃はBASICを独学で学んだというだけで，社内のパソコン係に抜擢されたという話さえ聞かれました．

BASICは形を変えて進化し，今ではVisual BasicやCA-Realizerのようなツールもあります．

● Pascal

Pascalは，AlgolとBasicの長所をあわせもち，使用する変数はすべて宣言するといった厳格さをもった構造化言語です．

Pascalは当初教育用言語として用いられましたが，パソコンの世界ではBorland社のTurbo Pascalの登場により実務にも使われるようになりました．

Pascalの直系言語としては，Modula-2やOberonなどがありますが，国内ではあまり使われておらず，特にヨーロッパで人気が高い言語です．またAdaもPascal系言語です．

● C/C++

Cについては説明するまでもないでしょう．Algol

の流れを引きMartin Richardsが開発したBCPL (Basic Combined Programming Language)をもとにして作られたB言語を親とします. 詳しくは,「プログラミング言語C(共立出版刊)」に書かれています.

しかし, Cのご先祖様のBCPLプログラムはなぜかあまりお目にかからないので, ここで紹介しましょう.

```
// hello.b - BCPL "hello world"

GET "LIBHDR"     // #include ヘッダー
                 // ファイル
LET start( ) BE  // main( )
$(               // {
  writes ("hello world.*n")
$)               // }
```

C++はB. Stroustrapが開発した言語ですが, Cにクラスなどのオブジェクト指向的な機能を加えたハイブリッドな言語です. なお, コメントが//であるのは, C++がBCPLの流儀にもどっただけであることがわかります.

● APL, SNOBOL

APLは数学用に開発された言語で, パソコンよりは大型計算機で使われることが多いようです. SNOBOLはテキスト処理用の言語で, おもに人文科学の分野で使われることが多いようです. APLもSNOBOLもIBM PC用のフリーソフトがあるので試してみることは簡単です.

● オブジェクト指向言語

最近脚光あびているオブジェクト指向言語も意外に歴史は古く, 1960年代のAlgolの特長を継承して, オブジェクト, クラス, メッセージを導入したSimulaにさかのぼります.

オブジェクト指向そのものはプログラミングのひとつの考え方なので, 言語にその機能があるからといってでき上がったプログラムがオブジェクト指向的なプログラムになるとはかぎりません.

今では, オブジェクト指向言語の数も多く, Smalltalkを筆頭に, ActorやEiffel, Lisp系にもLoops, CLOS, Flavorsなどがあります.

● Lisp

Lispはもともと人工知能の分野で使われた言語で, ほかのコンパイラ言語が事務処理の専門家や技術者によって使われるのに比べて異質です.

しかし, 従来のフォン・ノイマン言語の硬直さを解決するものとしてみれば, その貢献度は高いといえるでしょう.

これまでのプログラマの大半は, 与えられた命令を駆使することだけに慣れすぎているので, Lispの世界は途方もないものに感じられるようです.

● 各種テキスト処理/スクリプト言語

テキストフィルタといえば, UNIXのAWKやPerlが思いつきます.

最近ではコンピュータを設計する側は, ユーザーフレンドリなソフト/ハードを設計することに専念していますが, ある程度「コンピュータ慣れ」したユーザーは, コンピュータの思考に即した操作ができてしまいます.

本格的なUNIXのユーザーは, コマンドラインで自分の欲している情報をてきぱきと処理しているようです.

例えば,「ウクライナ共和国の首相はだれか」とか,「政府開発援助の受取り額がもっとも多い国は」といった場合, 最初の例では「『ウクライナ共和国』のデータベース(整列されたテキストでも十分)を見つけて,『首相』の項目を検索する」だけです. AWKやPerlを使えば, このようなことは簡単です.

しかし,「ウクライナ共和国の首相はだれか」を理解し, 検索してくれるコンピュータを研究することが, ユーザーフレンドリなコンピュータの研究であると言われ続けてきました. しかし, これはちょうどアンドロイド(人間とそっくりなロボット)の研究と似たもので, 実用になったという話はありません.

AWKやPerlを使えば, 短時間の労力で, はるかに効率的に結果を得ることができるのですから「ユーザーフレンドリ」にどれだけ分があるのか, テキスト処理言語が使える環境にいると疑問になってきます.

〈利根川　歩〉

PC98擁護論

1982年7月に初代PC9801が発売されて以来，PC98シリーズは圧倒的なシェアをキープしてきました．

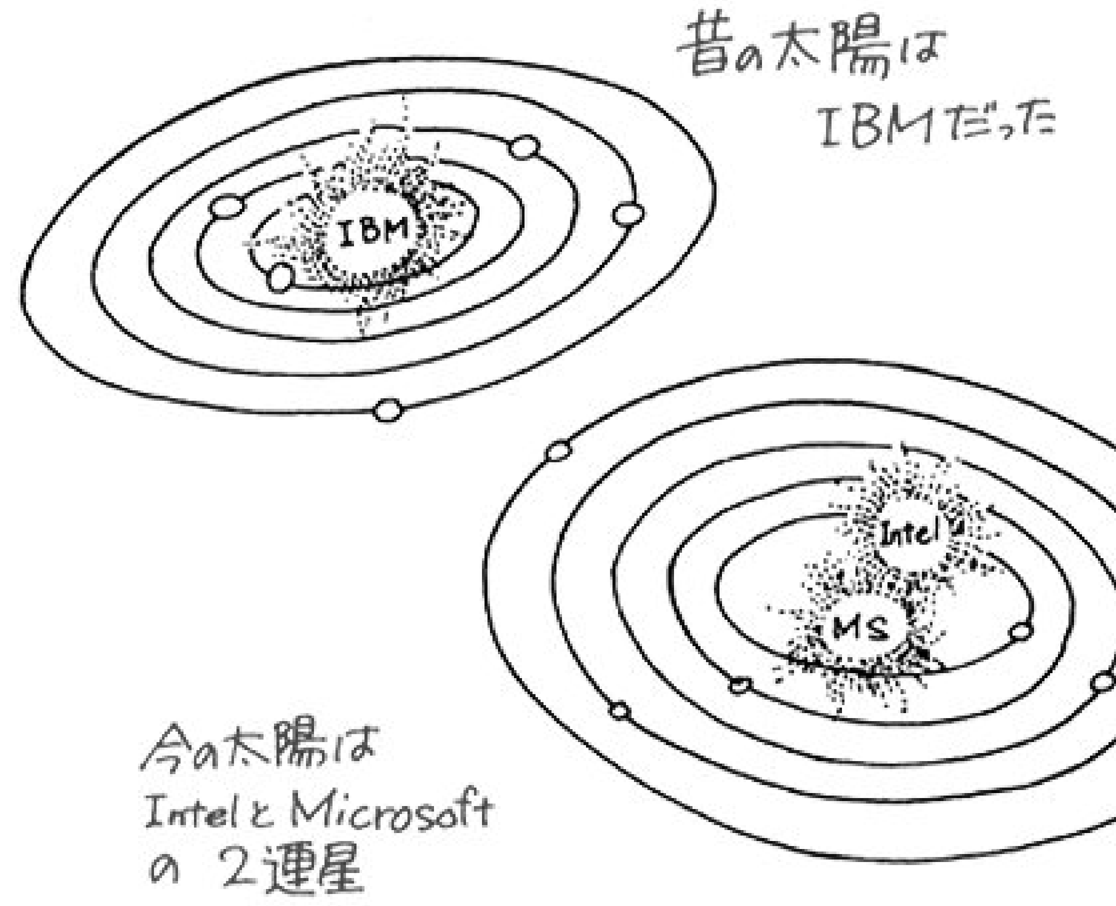
しかし，最近ではIBM PC(互換機)の追撃を受け，以前ほど優位な立場ではなくなってきました．

それは，DOS/V＋Windows 3.1が発売されてからPC98とIBM PCの機能上の差がなくなり，アプリケーションプログラムも双方で走るため，低価格なIBM PCに対して高いと感じるPC98の魅力がなくなってきたからと言えます．

パソコンの種類が増えたことは，ユーザーにとっては選択肢が増えて歓迎すべきことで，価格，性能，互換性，将来性などで好みの商品を選択できます．

しかし，初代PC9801(**写真1**)から使用している筆者としては何か残念で，また一部ではIBM PCの優位性ばかり強調され，PC98の欠点ばかりが論議されている場合が多いような気がします．

このような中で，筆者の独断と偏見でPC98の良さ/特徴などを，生い立ちを含めて紹介します．



PC98の生い立ち

● PC8001の登場

NECのPC98について語るには，1979年に発売されたPC8001という8ビットパソコンのシリーズと関連して話を進める必要があります．

PC8001は，その発売当時には国内で発表されていたボードコンピュータや一体型コンピュータを大きく越える完成度を持った製品でした．また，海外に目を向けても当時のPC8001を越えるものはあまりなかったと思います．

PC8001にはMicrosoft製のBASICインタプリタがROMで搭載されており，CPUにはIntel 8080上位互換でZ80互換であるNEC製のμPD780が採用されていました．



〈写真1〉PC9801

● パソコン発売時のコンピュータ業界

さて，パソコンが出始めた当時のコンピュータ業界に目を向けると，コンピュータは汎用機と呼ばれるメインフレームが中心でIBMが圧倒的に強力でした．

IBM以外のメーカーは，OSをはじめ何から何まで，すべてIBMと互換をとらないと売れず，IBMの新製品が出るとすぐに後を追って互換の機種を発表していました．

汎用機以外では，ミニコンやオフコンと呼ばれる機種があり，ミニコンは主として制御用に利用されていました．

パソコンを発表する前のNECは，IBM互換ではない汎用機を海外メーカーと提携し製造/販売していましたが，富士通や日立などの主力の国産メーカーはIBM互換でした．

NECはIBM非互換機でありながら，安くて，性能のよい中小型機で健闘していました．

筆者の独断では，国産のコンピュータメーカーのハードウェアは，低価格，高速で本家IBMに見劣りするものではありませんでした．

しかし，独自のOSではユーザーになかなか受け入れてもらえず，結局OS(システム)の基本仕様は業界での事実上の標準仕様を作成するIBMのあとを追随していくだけでした．

当時のこのようなコンピュータ業界の構造は，パソコンが主流になった現在でも変わっていません．昔のIBMがIntelとMicrosoftに変わっただけのような感じがします．

● 端末のNEC

このように，NECは汎用機や大型機ではふるいませんでしたが，中小型機，オンライン用端末では強く，ホストはIBM機でも端末はNECというシステムもありました．

端末は，オンラインシステムで遠隔地から電話回線を経由して操作するため，モデム接続用のRS-232C端子，CRT，キーボードそして全体の制御のためにCPUを持ち，内部構成はほとんどパソコンです．PC8001にも端末の思想は引き継がれています．

当時，オーディオ用カセットテープを外部記憶装置に代用していて，一般ユーザー用のモデムも数少ない時代に，RS-232C端子が標準で付いていました．

画面のアトリビュート属性も，普通では使わない端末系で使用するシークレット属性があり，ディップスイッチの切り替えで端末モードで立ち上がるなど，さすがにNECと思ったものです．

PC8001はすぐにPC8801シリーズとなり，一大ヒットをしました．そのような時期に初代98が発売されましたが，基本的には端末系の設計思想を引き継いだものです．例えば，RS-232CもIBM PCとは比較にならないほど，BIOSレベルでのサポートがきちんとしています．

また，PC98の発売時に，NECにはN5200というオフコン(端末)があり，CPUは8086で，専用OSのPTOS(ピートス)を持ち，当時から本格的マルチタスクでアプリケーションを動作させられるという本格的な製品でした．

また，NECはコンピュータでの上位互換，上位アップグレードに力を注ぎ，パソコンでも上位互換に努めました．一例をあげれば，PC98ではPC8801のBASICの互換およびPC8001のBASICの互換まで考えられています．

今日では，上位互換は当たり前のように考えられていますが，先を見越した設計が必要になり，メーカーの実力を問われるところです．PC8001当時のパソコンには，互換性に無頓着なメーカーもあり，新製品が出るたびに，以前のソフトが使用できない機種もありました．

このように，PC98は突然現れたものではなく，実際の業務で使用される端末の血を受け継ぐもので，PC8001からの実際の業務に使用できる完成されたマシンと言えるでしょう．

その設計のよさを示す証は，今でも初代PC98を使用できることです．初代PC98のときから高速な漢字表示を実現しています．現在では文字数が少ないとも言われる80字×25行ですが，いまでもりっぱに使用できます．

初代98の功績

国民機と呼んでいたメーカーもありましたが，ハードウェアの標準化と，膨大な動作可能ソフトを持ち，数多い人がそれなりに操作を習得したPC98はそれだけで一つの文化といえます．

パソコンの操作は車の運転と同じように，習得するにはそれなりの時間が必要です．特に，キーボードの配置，各種のコマンド，キー操作などは慣れとして，操作する人の体に染み付いています．

これを再び訓練して別の機種に慣れるようにするということは，膨大な社会的損失です．そういう意味では，パソコンのマンマシン・インターフェース自体に権利があり，その操作を他のプログラムが使用できないなどということは，重大な社会損失といえます．

PC98の基本スペックは，現在でもパソコンの必要最小限の仕様となりました．640×400ドット，80×25字で狭いとも言われる画面も，ノートパソコンなどでは十分使用できるスペックで，最初からこの仕様を採用しているところなどは先見の明があったと言えるでしょう．

また，PC98のキーボードの配置で育った人には，他機種のキーボードが操作しづらく，逆にPC98が標準キーボード配置になっているとも言えます．

コラム①● IBM互換汎用機びっくりの世界

金融，流通，行政などで行われている大量のデータ処理には汎用機が使われています．一般には目につきにくいのですが，多くの方がキャッシュカードや，座席予約などで知らず知らずのうちに使用しています．これらの汎用機には，IBMまたはIBM互換のものが多く用いられていています．

これらは，20年以上も前から同一のアーキテクチャを持ち続けており，上位互換で仕様を拡張してきました．したがって，つぎはぎ的な古い部分をあちこちに感じます．

例えば，CPUの命令体系にスタックがなく，サブルーチンコールには，リターンアドレスをレジスタからメモリにプログラマが待避させます．

またデータ参照のオフセットは12ビットしかなく，4096バイト以上ごとにベースレジスタを配置する必要があったりします．実際，アセンブラではこれらを意識してコーディングすることになります．

このように，今のパソコンからみるとずいぶ不便な感じがしますが，長期に渡る互換性の維持で，企業の高額なソフトウェア資産を支えてきました．

仕様の作成

● IBM PC の仕様は誰が決めるか

現在，IBM PC(互換機)と呼ばれているマシンはPC/AT が基本仕様です．しかし皮肉なことに，IBM の努力で互換を維持し続けたわけではありません．

IBM は PC/AT を発表したときに仕様を公開し，この結果，他社が低価格で高性能な互換製品を販売したため，互換機メーカーのほうが成長しました．

このため IBM は Microchannel というバスを搭載した PS/2 シリーズを発表し，ISA バス対応の従来からの互換機から市場を取り戻そうとしましたが，逆に Microchannel 搭載機が売れず，IBM 社自体も逆に ISA バスに戻っていきました．

このように，現在では IBM PC 互換機自体の仕様の決定権は IBM 社にはありません．また，特定のメーカーが仕様を決定できる状況ではありません．

● PC98 の仕様

PC98 では NEC が仕様を作成しています．IBM PC 用に出現した仕様も，いったん NEC に取り込まれ，PC98 に組み込まれています．

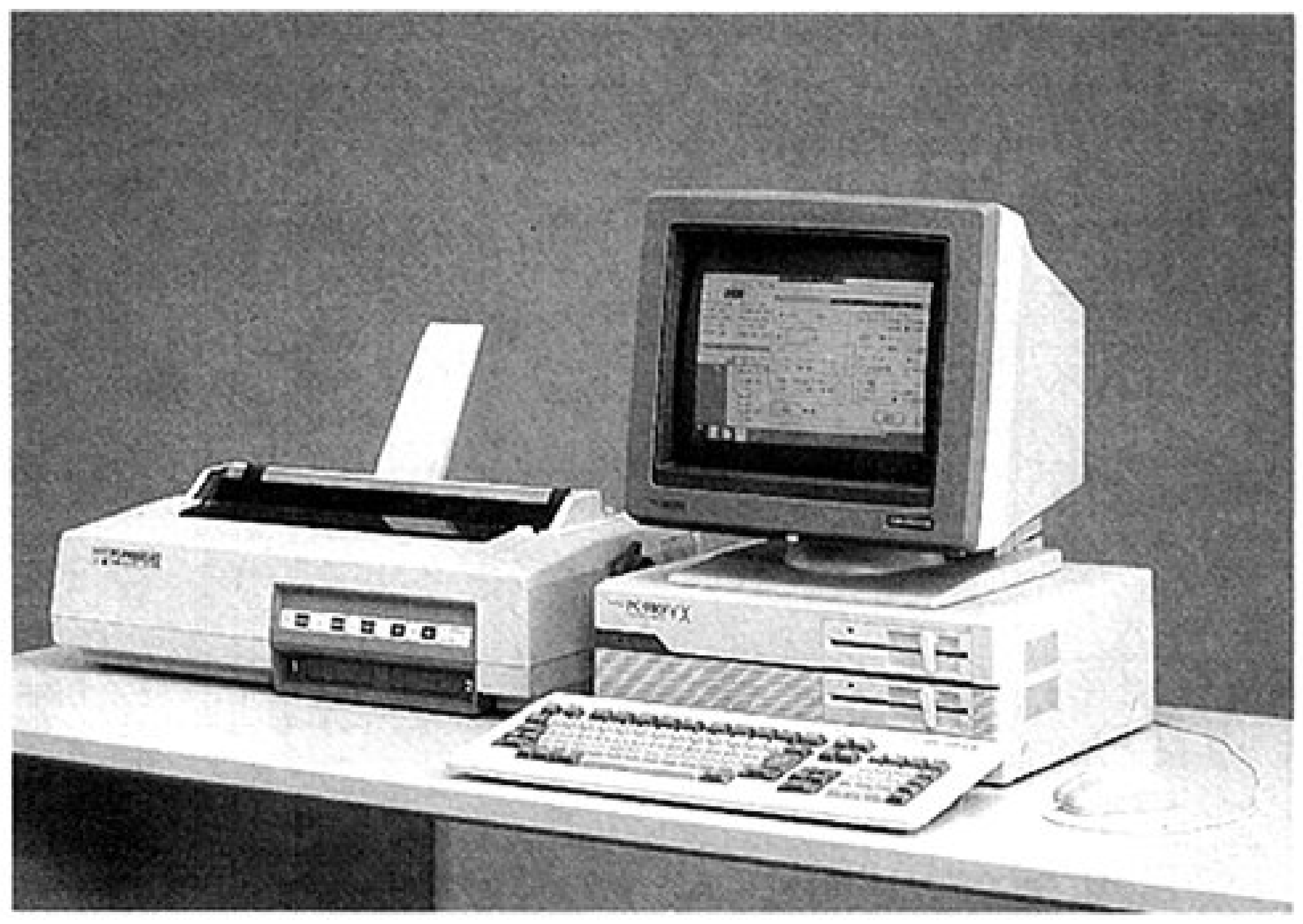
以前，NEC 独自の EMS 仕様が出現し，対応していないソフトが動作しないという問題がありました．

これは，EMS 仕様自体を 98 にアレンジして，空きメモリを増やすという意味があったのですが，英語版から日本語版に移植したソフトなどでは動作しないものが多く，最終的に NEC は従来と互換の EMS 仕様もサポートしました．

こういう問題もありましたが，最近では互換性のない仕様を採用するというようなことはなくなり，かえって一社で仕様を決めているため安定感があります．

また，NEC の新機種の成功は，市場で受け入れられるかどうかにかかってきます．早い話が，数多く売れた機械は標準仕様になりえるわけです(**写真 2**)．

PC98 シリーズも過去にはいろいろありました．ラップトップ LT，XA から始まったハイレゾ機，H シリーズなど，PC98 を発売して，国内パソコントップメーカーが十分に市場調査をして発表したはずの新機種でも，必ず受け入れられたわけではありません．



〈写真 2〉ベストセラー PC9801VX

ユーザーは目新しい機器に飛び移るより，互換性や価格などを吟味し，ずいぶん堅実であったように思います．

PC98 と IBM PC の拡張ボード上の機能

PC98 と IBM PC のハード構成で大きく異なるところは，拡張ボード上の機能でしょう．両者とも全体のハードウェア構成には大差ありませんが，IBM PC はマザーボード上ではなく拡張ボード上にほとんどの機能を載せ，必要に応じて交換や追加ができるようになっています．

初代 PC98 はフロッピディスク・インターフェースを標準でサポートしていましたが，PC9801E では拡張ボードによるサポートに変わりました．

当時は，2DD と 2HD の 2 種類のフロッピディスクがあったため，コストダウンを図るために，ユーザーがインターフェースの選択をできるようにしていたのです．しかし，PC9801E 以降の機種では 2DD/2HD 両方が読めるインターフェースになったため，マザーボード上に戻されました．

コラム② ● 昔もあった CPU の置換

初代 PC9801，PC9801E，PC9801F は CPU が 8086 ですが，V30 に置き換えて動作させることができます．

V30 は，NEC の 8086 互換 CPU で，PC9801VM などで使用されました．8080 のエミュレータを持ち，V30 で動作する CP/M80 のエミュレータなどもありました．

8086 とピン配置が同じで，そのまま差し替えでき，2～3 割のスピードアップが期待できます．

V30 発表当時，CPU を置き換えるだけで，こんなに簡単にスピードアップができるものだと感激したものでした．

そのため，より高速な互換の石が出ないかと期待していたところ，より高速な V33 の発表がありましたが，ピン互換ではなくがっかりしたものでした．

〈表 1〉 初代 98 のおもな周辺機器の価格

型名	内　　容	価格
PC9801	CPU	298,000円
PC8853N	14インチ・カラードットディスプレイ 640×400, ディジタル16色	168,000円
PC8882	8インチ・フロッピディスク・ドライブ 2基付き	400,000円
PC9801-01	JIS第一水準漢字ROMボード	40,000円
PC9801-03	オーディオカセット・インターフェースボード	15,000円
PC9801-02	128Kバイト増設メモリ	72,000円
PC9801-07	5インチ固定ディスク・インターフェースボード	20,000円
PC98H31	5Mバイト5インチ固定ディスク	478,000円
PC8886	8インチ・フロッピディスク10枚	198,000円

このように，マザーボード上に周辺機器用インターフェースが搭載されていることは現行機種まで変更がなく，必要機能であるとメーカーが判断した部分です．

現在では，周辺機器インターフェースをマザーボード上から追い出すメリットは，原価が下がることです．パソコン発売当時は周辺機器の値段が高価だったため(**表1**参照)，本体に内蔵せずオプションにし，また接続インターフェースを拡張インターフェースにしたのだと思います．

IBM PC では，RS-232C インターフェースから，プリンタインターフェース，ディスプレイまでを拡張ボード上で提供しました．これは，基本的スペックを設計時に決定せず，後送りにし，ユーザーの選択にまかせたともいえます．

PC98 と IBM PC の グラフィックの比較

C コンパイラに付属しているコンソール入出力ライブラリやグラフィックライブラリが充実してきたので，以前ほど画面表示やキー入力のために VRAM 操作や BIOS 呼び出しを行わなくなりました．

新旧の機種間の差異もライブラリが吸収してくれますし，PC98，DOS/V 両方に対応したライブラリを利用すると，機種をほとんど気にすることはなく非常に便利です．とは言っても，グラフィックで使用するのかテキストで使用するのかといった，基本的な構造は把握しておく必要があります．

また，TSR プログラムなどは，プログラムサイズの関係で自分で画面に書き込みなどを行う必要があります．

画面のハードウェアに関しては，IBM PC の場合 CGA，EGA，VGA と移ってきました．DOS/V 機では VGA 上位互換のディスプレイをもっています．

PC98 は，初代 98 からの 640×400 ドットのノーマル画面と，H98 などの 1120×750 ドットのハイレゾ画面があり，テキスト，グラフィックの重ね合わせの画面構成となっています．

また，PC9821 の 640×480 の拡張グラフィックに加えてウィンドウアクセラレータがあり，それぞれに細かい差異があります．

こうしてみると PC98 にもずいぶん種類があり，驚いてしまいます．しかし，ほとんどの場合は 640×400 ドットのノーマル画面をサポートすれば OK です．画面に凝らない(8色)なら，初代 98 のスペック(8色グラフィック画面1面)に合わせ，それ以外なら，初代 98，E，F，を切り捨て 4096 色中 16 色，グラフィック画面2面を使用すればよいでしょう．

プログラムに適切な初期化をして初代 98 に合わせておけば，ほとんどの 98 でノーマル画面は動作します．

いずれにせよ，機種に依存した部分をカバーする適切なライブラリを使用しないときは，プログラムでハードごとに処理を行います．

ハードディスクのパーティション

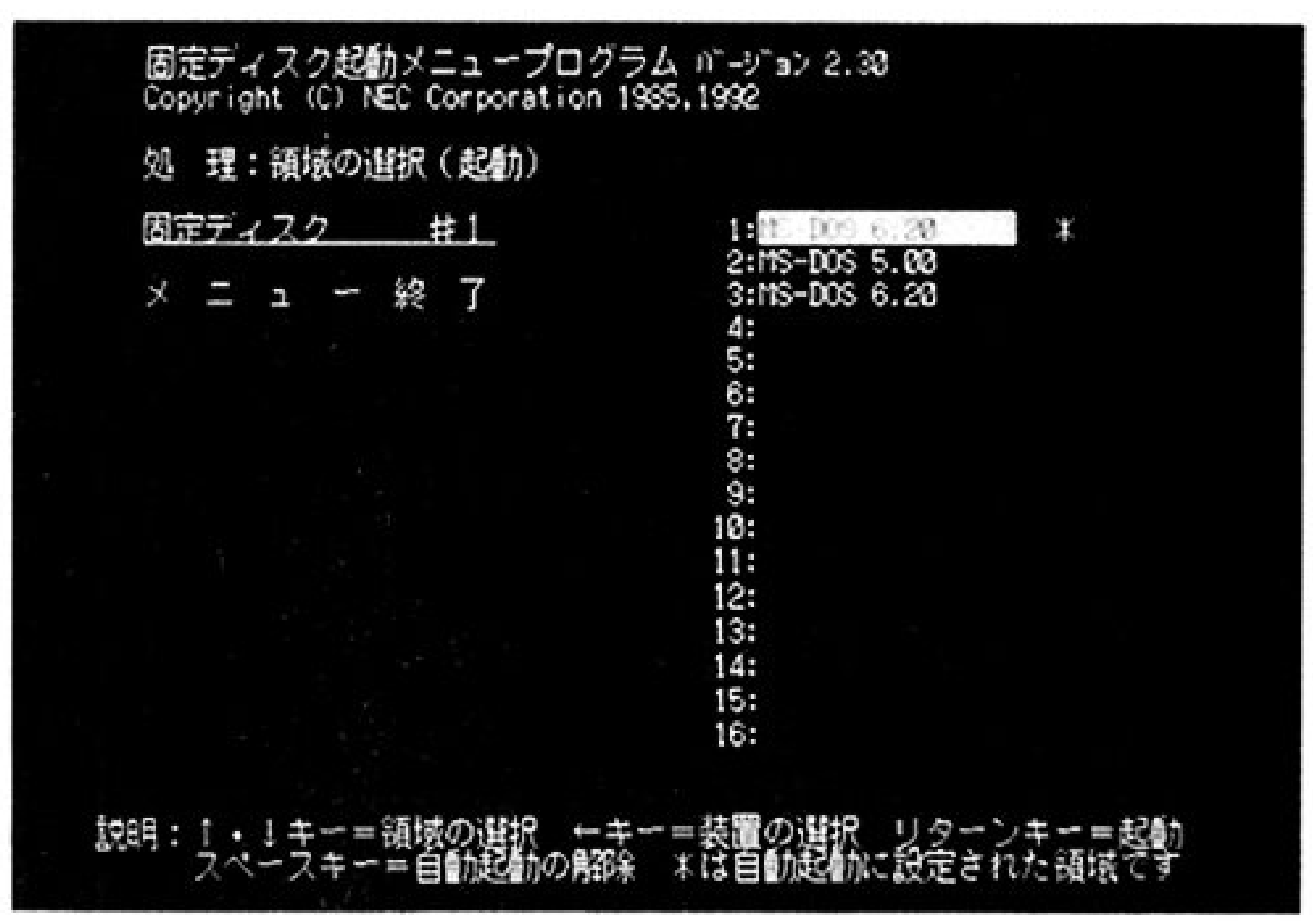
● PC98 のパーティション

Windows NT 3.5 を PC98 にインストールしたら，今まで表示していたリセット後の固定ディスクの起動メニューを表示しなくなり，困ってしまったことがありました．どうも，インストーラが自動起動モードにしていたみたいです．

このように，自動起動になっているシステムでは，リセット後の起動時に [TAB] キー(ハイレゾ時 [ESC] キー)を押すと固定ディスク起動メニューを表示します(**写真3**)．

起動メニューは，固定ディスクごとにパーティションの一覧を表示し，どのパーティションから起動するかを操作者に指定させます．また，＊を表示しているパーティションは，自動起動するパーティションです．

このように PC98 の場合，複数のドライブやパーティションから立ち上げができるようにシステムに組み



〈写真 3〉 PC98 の固定ディスク起動メニュー

込まれています．

また，PC98の場合，フロッピディスクで立ち上げた場合，**A:**がフロッピディスクに，ハードディスクで立ち上げた場合，**A:**はハードディスクになります．

PC98では，フロッピディスク2HD 4台，2DD 4台接続できましたので，このような仕様になっているのでしょう．

● IBM PCのパーティション

IBM PCにハードディスクがサポートされたのは，PC/XTからです．この頃の面倒なハードディスクの扱い方が今でも残っており，PC98ユーザーがもっとも戸惑うところでしょう．

▶基本的に一つのOSだけをサポートする

IBM PCでは，PC98の固定ディスクの起動メニューのようなものがなく，最近のようにNT，OS/2，Unixといった OSを起動時に切り換えて使用したい場合に困ります．

確かに，OS/2のブートマネージャやNTのブートローダなどで複数のOSを選択することも可能ですが，PC98のようにシステムレベルできちんとサポートされているものではないため，一つのパソコンにOS/2，NT，Windows95，Linuxをインストールしたい場合はなんらかの処置が必要です．



〈写真4〉IBM PCのハードディスクのフォーマット

▶フォーマット作業が面倒

PC/XTのハードディスクの物理フォーマットには，SYMDEBなどのデバッガを起動して拡張ROM BIOSの所定のアドレスをGOコマンドで指定するという方法を取っていました．

今では，これほど面倒な作業は必要ありませんが，それでもハードディスクの物理パラメータ(ヘッドやシリンダの数)の設定を求められるなど，PC98に比べてユーザーフレンドリとは言えません．

また，FDISKによるパーティションの設定とFOR-

コラム③●国産の486はなぜないのでしょうか

● 国産パソコンは日本製か

電子立国日本と言う言葉もありましたが，アジア諸国の追い上げ，円高，工場の海外移転，空洞化のなどの流れの中，コストアップへの対策のため，PC98の低価格機種も香港で生産されています．

ハイテク産業の代表のように思えるパソコンですが，本当に日本が作ったと言える機械はあるのでしょうか．

Macintoshは，誰が見ても米国の機械ですし，デザインからもいかにも米国的だと思います．また，アジアが生産の拠点になってしまったIBM PC(互換機)は，ほとんど国籍不明であるような感じがしますが，Intel主導のCPU，Microsoft主導のOSというように，米国の機械と言えるでしょう．

国民機PC98は，前述のとおり海外生産にシフトしつつありますが，CPUやOSなどの主要な構成品は米国製で，よく考えてみると日本製という言葉は適切ではないような気がします．

● パソコン用CPUを作らない理由

日本では486のようなCPUを作ることはできない(技術がない)のだ，と悲観的に言う人もいますが，本当にそうなのでしょうか．

メモリなどの生産実績でみると，半導体工場でCPUそのものを製造する力は十分にありそうですし，逆に得意としているようにも思えます．

また，大手電機メーカーはスーパーコンピュータを製造/販売しているわけですから，高速CPUのアーキテクチャの設計などもできないわけではないようです．

ではなぜ，となりますが，CPU以外の半導体に目を向けると，日本が輸入する必要のあるものは取り立ててないように思います．

つまり，メモリ市場などで国産メーカーは大きな売り上げをもっているため，米国から知的所有権などで裁判沙汰になるかもしれないようなCPUなどには手を出さないというわけです．

現在の日米間の状況から見ると，危ないものには手を出さず，できるだけ波風の立たないような液晶，メモリなどで利潤を上げることを考えているように思えます．

MAT による論理 FORMAT の二段階の操作なども，パソコンの初心者には設定不可能とまで言われています(**写真 4**).

▶ハードディスクを追加するとドライブ名が変わる

IBM PC では，ハードディスクは C ドライブからと決まっています．これ自体は問題ないのですが，例えば次のようなシステムがあったとします．

C ドライブ：IDE または ST506
D ドライブ：SCSI
E ドライブ：SCSI
F ドライブ：SCSI

このシステムに IDE ドライブを 1 台接続すると，次のようになってしまいます．

C ドライブ：IDE または ST506
D ドライブ：IDE
E ドライブ：SCSI
F ドライブ：SCSI
G ドライブ：SCSI

これでは，Windows などでは環境設定まで影響を及ぼすことがあるため，アプリケーションが起動しなくなったり，最悪の場合はシステムを再インストールすることになります．

また，このようなドライブ増設によるドライブ番号のずれは PC98 でも発生します．

IBM PC のフロッピディスクはなぜ遅いか

IBM PC 互換機のフロッピディスクのアクセスは，PC98 に比べると非常に遅く感じられます．同じ FAT を使用しているので，この点では明らかに PC98 のほうがよくできていると言えるのではないでしょうか．

そこで，なぜ IBM PC のフロッピディスクは遅いかを筆者なりにまとめてみました．

なお，内容については，筆者の調査，認識不足のため一部推測している部分がありますが，こういう意見もあるということでご了承ください．

● ハード構成

IBM PC も PC98 もフロッピディスク・コントローラに NEC の μPD765 互換 IC を使用しており，DMA や割り込みなどについては大差ありません．

ただし，PC98 のフロッピディスク BIOS は μPD765 のほとんどの機能をサポートしています．IBM PC では，通常の READ/WRITE，フォーマットをサポートする程度です．

フロッピディスク・ドライブのモータ制御は，IBM PC の場合，通常は停止させておき，必要な時だけ回転させます．接触したままでモータを回転させると媒体が摩耗するので，必要な時以外は回転を止めているのだと思われます．

PC98 の場合は，基本的に常時回転していますが，3.5 インチ・フロッピディスク・ドライブではモータの ON/OFF 制御を選択できるようになっています．

PC98 の場合は回転が一定に達すると，フロッピディスクの読み書き可能状態で READY 信号が ON になります．この READY 信号線は，μPD765 の READY 端子に接続されています．

IBM PC では READY を使用せず，μPD765 の READY 端子は常に READY 状態に固定されています．また，IBM PC では DISK CHANGE 信号を使用しています．DISK CHANGE 信号はフロッピディスクが取り出されたことを示す信号で，SEEK させると (STEP 信号の変化) リセットされ，現在の状態を示します．

● 回転の立ち上がり検出

PC98 のフロッピディスク・ドライブには READY 端子があり，定速になると READY 信号が ON になります．

IBM PC では，フロッピディスクにアクセスした場

コラム④●設計思想と企業戦略

DOS/V 機を自作した方ならご存知のとおり，IBM PC では各パーツが規格化されています．

このことは，後々性能的に劣る部分を交換することで，同じパソコンを継続して使えることを意味します．

PC98 ユーザーにとっては非常にうらやましいことで，例えば古い PC98 のユーザーは，マザーボードだけを最新機種のものに交換できたらと思うことでしょう．

さて，このことを PC98 の基本設計のまずさと指摘する方がいます．

しかし，これは基本設計のまずさと言うより，実際のところ企業戦略と言うべきです．

パソコンに限らず，一般にメーカーの商品企画では下位機種と上位機種の差別化を図ります．

そして，一通りの品揃えを行い，高く売れるものは高く売る．また安価な下位機種に上位機種の客が流れないような処置をするのが通例です．

合，最初はモータが停止しているので，BIOSはまずモータを回転させ，一定速度に達するまで待ちます．

これは，μPD765のREADY端子が常にREADY状態であるように端子に固定的な電圧をかけてあるので，実際にREADY状態かどうかはソフトウェアでモータのONから一定時間を計測しているためです．

このため，昔のベルトドライブ方式で立ち上がりが遅いドライブも，ダイレクトドライブの立ち上がりの速いドライブもタイマを用いて一定時間待つしくみです．

PC98では，立ち上がりの速いドライブでは，すぐにREADY状態になりアクセスできます．

IBM PCでは，基本的にこのタイマは遅いままです．最近の一体型パソコンやノートなどでは，この時間は調整されているかもしれませんが，市販されているマザーボードのほとんどは，初期からの遅い状態になっていると思います．

● 読み出し中のフロッピの取り出し

IBM PCの場合，READY端子を固定しているために，アクセス中にフロッピディスクを取り出してもすぐに検出できません．

最悪の場合，μPD765はIDを探す状態でタイムアウトまで待ちになります．

● フロッピディスク・ドライブのREADY信号チェック

フロッピディスクをアクセスする場合，メディアが挿入されているかどうかをチェックしなければなりません．

PC98の場合，REDAY信号をチェックすればOKですが，IBM PCではDISK CHANGE信号をチェックします．

DISK CHANGE信号は一度シークをさせた後にチェックするので，少し時間がかかります．IBM PCのフロッピでアクセス時に短い「ガガッ」という音を発生する場合がありますがこれは，このシークの音です．

● 媒体エラーの場合

PC98の場合BIOSでの読み書き時にエラーが発生した場合，規定値回数リトライを行って呼び出し元に戻ります．

IBM PCの場合，エラーが発生するとBIOSがサポートする各フォーマット(2DD, 2HDなど)にモードを変え，そのモードで読み出せるかをリトライします．

ですから，エラーの媒体にアクセスすると，エラー処理に時間がかかる場合があります．

● BIOSの使い勝手

前述のとおり，PC98のフロッピディスクBIOSは，μPD765の機能をほぼ網羅しています．

ですから，自分でμPD765を操作することはあまりないと思います．IBM PCの場合，フロッピディスクBIOSでサポートされている機能が少なく，ちょっと何かしようとすると，μPD765を操作する必要があります．

当然DMAから割り込みまで操作するので，手軽にはできない部分があります．　〈WOK〉

〈p.80へつづく〉

コラム⑤● PC98の今後

IBMが開発したIBM PCは，今ではIBMの手から離れて世界標準のパソコンになっています．CPUはIntelの影響力が強力ですし，OSも圧倒的にMicrosoftが強力です．

バスに関しては，IBMの提唱したMicro channelが不調に終わり，VLバス，PCIバスが主流になっています．グラフィックアクセレータもまたしかりです．

IBM PC(IBM互換機市場)は，IBM PC以外のコンピュータメーカーを飲み込もうとしているように思えます．

一部で熱狂的なファンを獲得しているAppleのMacintoshはもちろんのこと，クライアント/サーバーコンピューティングの台頭で，IBM互換汎用機(パソコンの世界にいると気がつきにくいが，IBMは汎用機では絶対的な存在)の世界にまで影響を与えています．

このような現状を見ると，NECがいつPC98を放棄してIBM PC互換機路線に移るのかが野次馬的話題になります．

ハードウェア自体は，DOS/V機との共通化(IDEディスク，メモリのSIMM化，PCIバス対応)を図っていますが，特にDOS/Vと共通設計を行うと言う意味ではなく，低コスト化の一貫と見るべきではないでしょうか．また，NECは海外でNECブランドのIBM PCを発売しています．

NECは汎用機全盛期においてもIBM互換をとらず，独自路線を通して，それなりに善戦してきた経緯があります．そうすると，パソコンでも独自路線をとっていくと見るのが妥当なところのようです．

原　信一郎

Appendix

UNIX系スクリプト言語 AWKとPerlをマスターしよう

ちょっと乱暴ですが，UNIXといえばテキスト処理，テキスト処理といえばフィルタだといえるでしょう．

UNIXではgrep，sort，sedなど，いろいろなテキストフィルタがあります．

複雑な仕事もこれらのフィルタをパイプで組み合わせてこなすのがUNIX流だったわけですが，さすがに単機能のフィルタだけでは不便なので，より高機能のプログラマブルなソフトウェアが作成されました．その代表格がAWKとPerlです．

この二つは極めて優秀な「言語」であり，テキストデータを扱うバッチ処理において，現在,Cなどのコンパイラでプログラムを作成することは，ほとんど考えられません．AWKやPerlを使えば何分の一かの労力ですんでしまうからです．

また，AWKとPerlは，MS-DOS，OS/2，Windows-NTに移植され，そしてなんとMacの上でも動いています．

したがって，AWK，Perlは異なるOSを越えてひとつの仕事をする場合にも重要な手段となっています．

ただ，Windowsとだけは相性が悪かったのですが，BOW(BSD on Windows)の登場である程度の解決をみました．

AWKかPerlか?

よく筆者は「初めて学ぶとしたらAWKかPerlか？」という質問を受けますが，これは世の中で最も難しい質問のひとつではないでしょうか．

一応，AWKはコンパクトでやさしく，Perlは高機能，大規模とはいえます．

しかし，それぞれに独特の強い個性があるので一概に比較できません．「全く質が違うので比べられない」と断言してしまえばよいのですが，比較できる部分もあります．その日の陽気によって答えが変わってしまっても無責任なので，筆者は偶数日はAWK，奇数日はPerlと言うことにしました．

sed

sed(せど)は，edという非常にシンプルなエディタを元に作られました．この名前はStream EDitorの略です．流れてきた文字列に対する「自動編集機」といった感じでしょうか．

AWKもPerlもsedを祖としているといっていいでしょう．すなわち，「正規表現」と呼ばれる文字列のパターンを捉えるメカニズムをもち，それを用いて流れてくる文字列を加工するというのが本来の仕事です．

しかし，sedのスクリプティングはパズルめいていて特殊な技能を要求され，それはそれで面白いのですが，ループやジャンプなどの制御構造を用いるようなプログラムを自分で組むとしたら，やはりAWKかPerlということになるでしょう．

AWK

AWK(おうく)は，コンピュータ科学方面ではそれぞれに「超」有名な3人，エイホ，ワインバーガー，カーニハンによって作られました．AWKはこの人たちの頭文字をとったものです．

C言語とほぼ同様な制御構文や，連想配列という独特な変数を装備しています．1985年に関数定義の実装などで機能強化され，立派な「構造化言語」となりました．

それではAWKの紹介として，住所録の検索スクリプトを書いてみます．

リスト1のように「読み,氏名,電話番号,住所」が各行に書かれた**addr_dat.txt**というテキストファイルがあるとしましょう．

リスト1のファイルから，例えば読みが**murayama**である人の情報を取り出すには次のようにコマンドを打ちます．

```
>jgawk -f addr.awk murayama addr_dat
```

ここで**-f**オプションで指定された**addr.awk**というのがスクリプトなのですが，内容は**リスト2**のようなものです．

AWKスクリプトの基本型は，

〈リスト1〉サンプル・テキストファイル addr_dat.txt

```
...
murayama tomiiti,村山富市,03-1234-5678,首相官邸
hata tsutomu,羽田孜,03-0123-4567,前首相宿舎
hosokawa morihiro,細川護熙,03-0012-3456,細川城天主閣
...
```

〈リスト2〉
murayamaを取り出すスクリプト
addr.awk

```
BEGIN {
    FS = ","               # データが , で区切られている事を宣言
    pat = ARGV[1]          # murayama を変数 pat に格納
    ARGV[1] = ""           # murayama を入力ファイル名リストから除外
}
# 以下, addr_dat の各行に対する処理
$1 ~ pat {                 # 第1欄が murayama という文字列を含むとき以下を実行
    print "読み：", $1
    print "氏名：", $2
    print "電話：", $3
    print "住所：", $4
}
```

パターン { アクション }

です．**リスト2**では**$1 ~ pat**がパターンであり，それ以下の**print**文がアクションです．

AWKは入力ファイルを1行ずつ読み込みながら，各行を , で区切って変数**$1, $2,** … に分解しつつ処理を進めます．

このように機能をテキストデータを加工することに特化しているので，簡単にそして自然に住所録の検索ソフトウェアが作れてしまうのです．

そしてまた，住所録のメインテナンス，例えば新しい住所リストとのマージ，重複データの検出，一覧表の作成なども，AWKを使えば最小限の手間だけで行えます．

●『プログラミング言語AWK』

awker(AWK使い)の多くは，この本を読んでAWKに惚れ込んでしまった経験があります．この本の著者はAWKの作者たちですが，内容はAWKの簡単なスクリプト集にして仕様書であり，さらに実践的なAWKの応用も含まれています．それだけではありません．

アルゴリズムとは，またプログラムとは，そもそもコンピュータとはどういう機械か，などのいろいろなテーマをAWKを使って具体的に考えさせられる，実に切れ味のいい見事な例題が山盛りです．

この本を読むためにAWKを学んでも損ではないでしょう．

Perl

PerlはL.ウォールによって作られました．やはりUNIXユーザーなら同氏の作ったpatchコマンドを知らない人はいないでしょう．

ものすごいスピードでマイナーバージョンアップをしていましたが，昨年ついにバージョン5(Perl5と言われる)になりました．

「Perl」も何かの略称でしたが，今では覚えている人もあまりいないと思われます．Perlの特徴を一言でいえば，AWKの5倍か10倍か，あるいはそれ以上かもしれませんが，多機能であるということでしょう．

ファイル/ディレクトリ操作やシステムコール，ネットワーク機能まで備えているので，Perlひとつあれば，binディレクトリのすべてのファイルを削除してもかまわないと豪語する人もいるほどです．

本当はPerlをテキストフィルタと呼ぶことをやめればいいのですが，それにしてはテキストフィルタとしての能力が高すぎます．

AWKの機能を完全に含んでいるので，AWKで書いたスクリプトは，簡単にPerlスクリプトに書き換え

〈リスト3〉
リスト2をPerlに
置き換えたスクリプト
addr.pl

```
$pat = shift;           # murayama を変数 $pat に格納
while (<>) {
# 以下, addr_dat の各行に対する処理
    chop;               # 行末の改行を取る
    if (/$pat/) {       # murayama という文字列を含むとき以下を実行
        ($yomi, $name, $tele, $addr) = split(/,/); # データを , で分割
        print "読み: $yomi¥n";
        print "氏名: $name¥n";
        print "電話: $tele¥n";
        print "住所: $addr¥n";
    }
}
```

〈リスト4〉超高機能電卓 calc.pl

```
while(<>) { print eval($_)."¥n"; }
```

ることができます(a2pというAWKスクリプトからPerlスクリプトへの変換フィルタさえあるぐらいですから).

例えば,**リスト2**をPerlで書き換えると**リスト3**になります.

>jperl addr.pl murayama addr_dat.txt

と実行します.

AWKと比べると**while (<>)**, **chop**, **split**などが余分に必要となっていますが,それだけ汎用性が高いことを示しています.

また,それらを使わないで書く方法もあります.Perlでは同じことをさせるのに何通りもの方法が思いつき,それがこの言語の仕様の特徴のひとつでもあるのです.最新のバージョンのPerlは,データ構造の多重化,「オブジェクト指向」なども取り入れてたいへんなことになってきました.

Perlの設計思想は,基本的な能力を損なわない限り徹底的に機能を拡張する,ということではないでしょうか.

AWKがテキストファイル処理のために必要十分な極小の機能を持とうとしているのに対し,Perlは自己を最大に拡張しようとしているのにもかかわらず,破綻するどころか小回りが効いて抜群の使いやすさを保持しているところが奇跡的です.

● Perlらしいスクリプト

最後にひとつPerlらしいスクリプトをお目にかけたいところですが,どんなスクリプトもPerlの特徴を表すには偏っていて,不適切となってしまいそうです.いっそあえて思い切り特殊な例を**リスト4**に示します.Perlで電卓は,これでおしまいです.これで超高機能な電卓になります.これでは中途半端な電卓は使いたくなくなりますね.

<実行例>

>jperl calc.pl
355/113 ← 単純な計算
3.14159292035398221
$a=3; $b=4; ← 変数への代入
4
sqrt ($a*$a + $b*$b)
← 組み込み関数平方根を使う
5
sub f1{$x**3 − 3*$x +1;} ← 関数定義
f1=x^3−3x+1
$x=2; &f1; ← x=2の時のf1は?
3
while (++$i<=10){$s += $i;} $s;
← 制御文も使える
55
unlink ('awk') ← ファイルawkを削除
0
exit ← 終了
>

●「キャメルブック」

Perlについては,Perlの作者らによる――なぜか「バイブル」と呼ばれたりしていない,かわいらしく「らくだ本」と俗称される本――『Perlプログラミング』があります.

大部ですが,Perlの勉強のためには欠かすことのできない本です.Perlの面白さをたっぷり味わえること請け合いです.

処理系

gawk(GNU版AWK)とperlは,GNUのコピーレフトに準じた,いわゆるフリーウェアです.jgawk,jperlという漢字(2バイト文字)を意識した動作をする版も作られているので,みなさんも是非一度動かしてみてください.

◆ 参考文献 ◆

(1) エイホ/カーニハン/ワインバーガー;足立高徳 訳,プログラミング言語AWK,トッパン.
(2) L. Wall/R. L. Schwartz;近藤嘉雪 訳,Perlプログラミング,ソフトバンク.

特設記事

コンピュータサイエンスの概念の宝庫！

プログラミングの世界を広げる LISPをマスタしよう

岡田好一

ちょっと気になるLISPの世界

LISPはどんな言語か

初期のLISPは1960年頃に開発されたということですから，40年近い歴史を持つことになります．

しかし，日本でLISPが普通のプログラマにも扱えるようになったのは，一昔前の人工知能ブームのおかげだと筆者は考えています．

ちょうど1980年頃，米国では商用をはっきりと意識したLISP専用マシンが発売されています．また，使って楽しい，誰にも安心して使えるLISPはこの頃に生まれたといってもよいでしょう．その代表が有名なCommon LISPです．

注意しなければならないことは，第五世代コンピュータ計画において，生産性を飛躍的に向上させた，「普通に使われる」コンピュータを目指したのと同じように，LISPも文書処理や数値計算，グラフィックスを含むあらゆる計算機で使われるように努力をしていた，ということです(図1)．

現在，LISPは少し揺り戻しがあり，ふたたび研究者の言語になりつつあるのが気になります．しかし，かつてのような方言乱立状態はなくなりましたからScheme(スキーム)と呼ばれるLISPの一方言を知っていればよいようです．

いまLISPを知る利点

LISPは，実用という点では低迷している言語といえるでしょう．にもかかわらず，ごく普通のプログラマにも学習のしがいのある言語です．もし，時間に余裕があるのなら，ぜひLISPを導入し，触ってみることをお薦めします．LISPを知ると，計算機の面白さが倍加します．

● LISP用語に強くなる

例えば，X-WindowやOSF/Motifには，caddr型というものが出てきます．Windowsでも，アトムという用語が出てきます．これらは，LISPを起源としています．

LISP用語の詳細を知らなくても，とにかくプログラミングすることは可能です．しかし，内容を知っていると何となく安心ですし，インスピレーションが働くので，応用範囲が広がるかもしれません．

例えば，bit誌の記事に，

「consの90%はcarもcdrもとられずにゴミになる」という記述がありました[(1)]．

これをC++語に翻訳すると，「自動生成されたオブジェクトの90%はメンバーが参照されることなく，自動消滅する」となります．

それでは，C++でもオブジェクトの追跡プログラムを作って，パフォーマンスチェックをしてみようかな，という発想につながるでしょう．

▶ LISP用語が使われる理由

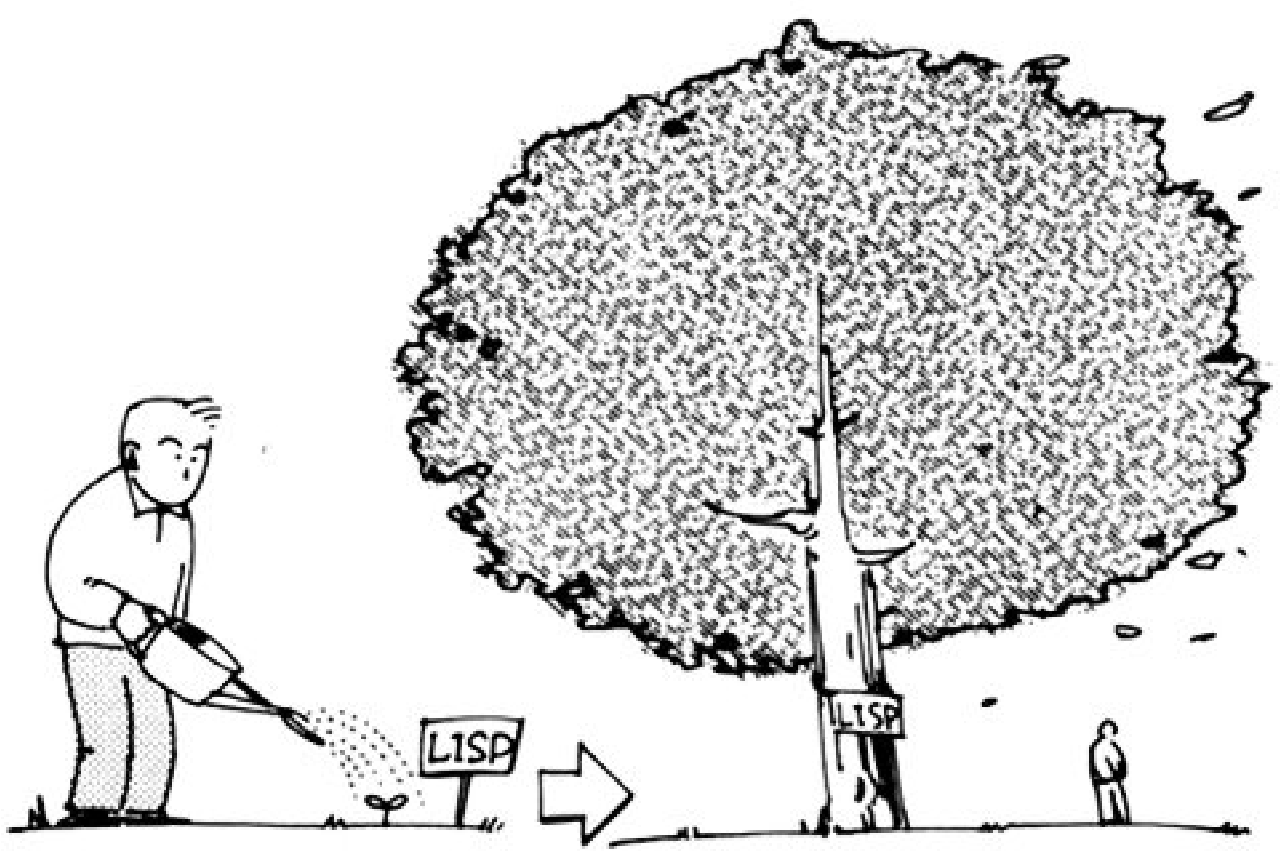
LISPを学んだ人々が，上記のウィンドウシステムなどを作り上げたのでしょう．

LISPを学ぶと，OSなどのシステムプログラミングのセンスが身につきます．これは，たいていのLISPの

〈図1〉LISPは人工知能専用言語ではない



〈図 2〉 LISP は育てるシステム(昔は…)



教科書が，非常に単純な LISP のカーネルを元にして，少しずつプログラミング環境を整えて行くように書かれているからだと思います．

つまり，LISP のスタイルは，問題に応じて，言語仕様を次々に拡大していく感じとなります．単純な LISP システムが，各種ユーティリティの整った，プログラミング環境に育っていくのです(図 2)．

この過程は通常のアプリケーションのシステム設計，つまり最初に要求があって問題分析し，仕様を確定し，設計/プログラミングしていく方向とはまったく逆です．プログラミングが，新たな問題を発見し，問題を確定させる方向です．

システム作りには，このような拡大の方向性が必要なのでしょう．もちろん，通常のシステム設計に LISP を使ってもルール違反ではありません．

● **プログラミングの幅か広がる**

筆者は，まさにこの理由で LISP の学習をお薦めします．極端に言ってしまえば，LISP が本格的に使えなくてもよいのです(使える方がいいです)．

シンボル(記号)の概念，リストの柔軟性，データ型が変数ではなく，データの属性であること，動的なデータ確保，ガベージコレクション，コンピュータサイエンスの概念の宝庫であることがわかります．

▶記号処理

LISP は，時に記号処理言語と呼ばれます．LISP における「記号」は，後ほど具体的な定義を述べます．しかしここでは，本来の意味，

　世の中の森羅万象を指すもの＝記号(図 3)

だと考えてください．

記号は，具体的なもの，ある人やある 1 個のリンゴも指せますし，ヒトというもの，リンゴというもの，といったもっと抽象的な区別/分類の概念も扱えます．波や風など，現象も記述できます．

ただし，LISP でやるように，記号に文字を使ってしまうと，量の表現はちょっと苦手になります．

こうした記号が，LISP の処理の主な対象(つまりデータ)です．具体例を挙げましょう．

okada-taro

は，ある人を指すでしょう．

apple

は，リンゴ一般を指します．

red

は，赤を指します．

なんだ，単なる文字列ではないか，と思う方も多いでしょう．そのとおりです．

しかし，意味やイメージが記号には付属していることがわかりますか．つまり，「red が青だ」と直感する人はあまりいないということです．

記号を決めていくと，互いに関連のあることがわかります．この関連を LISP ではリストで表現します．

例えば，

(color apple red)

は「リンゴが赤い」という事実を表し，

(father okada-taro okada-hikozaemon)

は，父子の関係を表します．

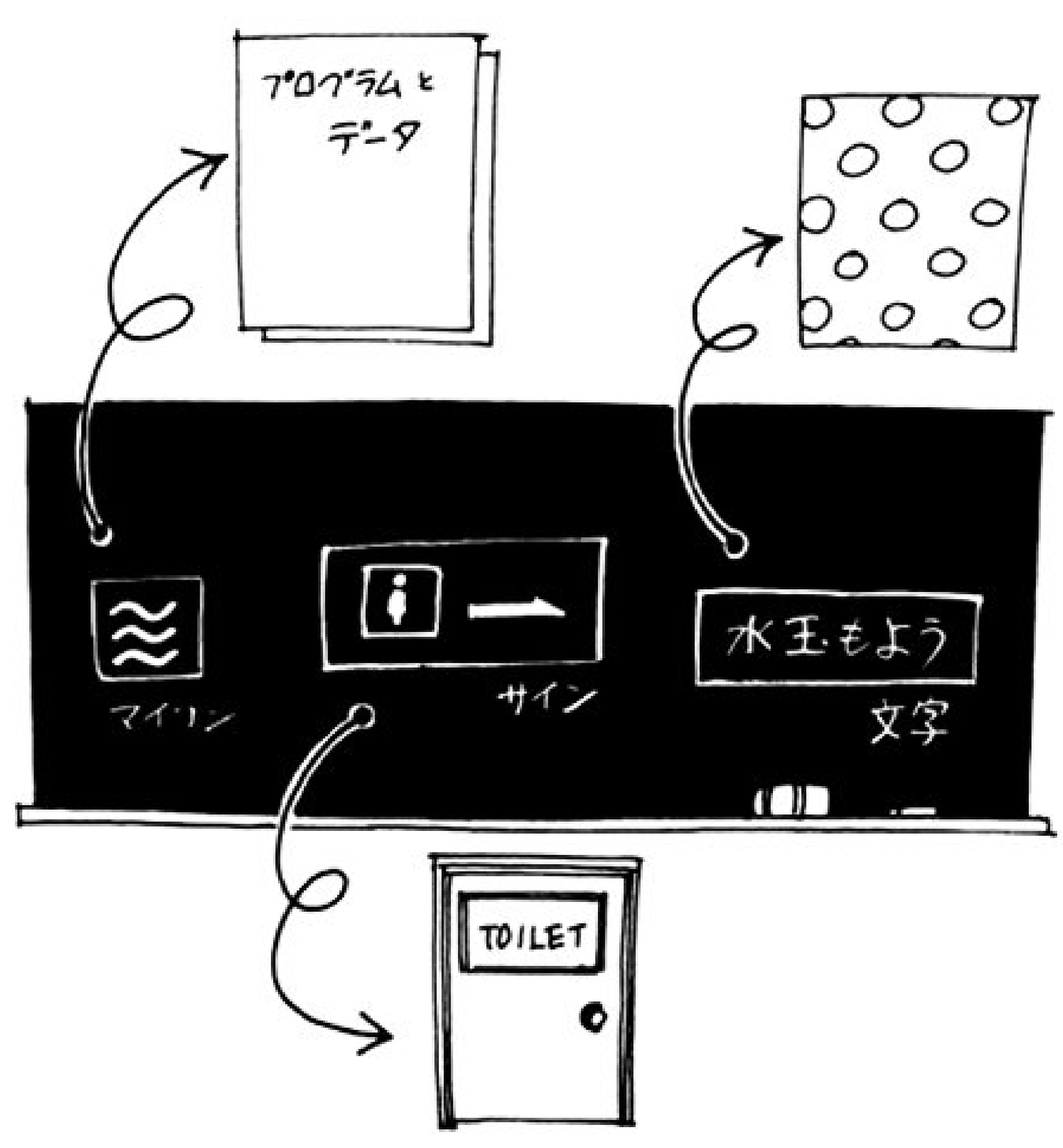
なんだ，記号を並べただけではないか，もっとはっきり言うと，データが文字列のレコード(あるいは構造体)ではないかと，また言われそうです．そのとおりです．

表現の方法はもともと存在するのでなく，上記の形で記号同士の関係を「表現」しているだけです．

表現方法はさまざまであり，別の記号の体系では，全然別の表し方となるでしょう．

もし，ある記号の関係を与えて，別の関係が導出されたらどうでしょう．何らかの計算が行われたことになります．これが記号処理です．

〈図 3〉 記号



▶記号処理の威力

具体的なイメージは教科書を学んでいただくことになりますが，LISPでは記号処理が簡単にできるので，コンパイラや数式処理がぐっと身近になります．例えCを使うにせよ，LISPを知っていると小難しい理屈抜きでどんどんプログラミングできるようになります．

LISPを使うとどれくらいうれしいかですが，筆者は約200行で簡単なPASCAL風言語のコンパイラを書いたことがあります．

このコンパイラは，ポケットコンピュータのLISPで稼働していました．ただし，セルフコンパイラではないので，コンパイル後のプログラムを実際に動かすには，MS-DOSとMASMが必要でしたが，プロが作るともっと少ない行数で，とりあえずのコンパイラが完成すると言われています．

同じコンパイラをCで書くと，何行になるのか見当もつきません．

● LISPではあたりまえ

▶多倍長整数と分数

LISPは数式処理の需要があるためか，多倍長整数(bignumと呼ばれています)のないLISPは少数です．例えば，120の階乗を最後の1桁までけなげに計算します．いかにもこれが計算機だ，という気持になります．

分数は，多倍長整数を二つ組み合わせます．分数の分子と分母の桁数は，計算の途中で案外大きくなるものです．

いかなる固定長整数でも，いずれあふれると言われています．ですから，無限桁数の整数，実用的には多倍長の整数が必要です．

▶数値計算と文字列処理

数値計算や文字列処理の能力は，BASIC相当だと考えてよいでしょう．浮動小数点数や配列，文字列は用意されていることが多いと思います．

なぜかLISPの教科書ではさらりと述べられますが，数値計算が特に不得意だ，ということはありません．文字列についても同様に，LISPだから不利になることはないはずです．

〈図4〉パターンマッチとバックトラック



ファイル処理も，LISP固有の機能を除けば，通常の言語と同じです．

コンパイラもありますし，インタプリタといっても，後述するように，中間言語コンパイラを常時使っているようなものです．現在のパソコンCPUを使うと，これがあの遅い遅いと言われたLISPかと思えるほど，ものすごい速度で計算します．

そして，上述の記号処理がからむと，他の言語に比べた生産性は飛躍的に高まります．

▶ハッシュ表など

記号処理では，頻繁に，関係を表す「表」というデータ構造を用います．表の実現にはいくつかの方法があります．

Common LISPには，自動的に大きさが拡大するビルトインのハッシュ表型があり，効率，使いやすさともに申し分ないと思います．

なお，記号処理が得意といっても，パターンマッチングとそれに関連して，バックトラック(図4)はプログラミングしなければなりません(言語にビルトインされていません)．LISPを使った人工知能関連の本では，パターンマッチングのプログラミング方法が延々と出てきます．

パターンマッチングは，DOSコマンドのワイルドカード(＊と？)を思い浮かべればよく，それを文字列ばかりでなく，一般の構造体(LISPではリスト)に拡張したものと考えてください．

バックトラックは，要するに試行錯誤を指すプログラミング用語です．パターンマッチングでは途中まで正解でも，その先の都合によって少し立ち戻る必要が出てきます．ゲームを解くプログラムなどに一般に見られるパターンです．この立ち戻りがバックトラックです．

とはいえ，計算機におけるたいていの処理パターンのプログラミングは容易です．現在では，パソコンでもOSを買うだけで，さまざまなユーティリティが使えます．しかし，余裕があるなら，LISPを1本インストールしておくと，気分的に安心です．

LISP購入ガイド

LISPの学習にはLISPの実物が必要です．教科書だけだと，なぜ別の方法で書かないのかが永遠の謎になってしまいます．

本稿を書くために雑誌の広告などを調べた時点では，LISP処理系の少なさにいささかがっかりしました．しかし，実際に集めてみると，どれもよくできたLISPで心配ありません．



〈写真１〉TUTScheme

ここでは，たまたま今回集めたLISP処理系について，筆者なりに特徴を述べてみます．

● **TUTScheme**(岩波ソフトウェアライブラリ，Scheme)(**写真１**)

日本製のLISPです．MS-DOSで動きます．安価でコンパクトなのに，関数の取り揃えに欠陥がないので，実用にも十分でしょう(いかにも日本製のLISPです)．

数値データには，多倍長整数と実数(32ビット)があります．文字列は，8191文字まで．配列型が用意されています．

つまり，まず高速BASICとして利用可能で(書き方はLISPですが)，プラスSchemeの勉強ができます．マクロが付いているので，LISPの強力さも実感できます．

● **muLISP-90**(**写真２**)

筆者が初めて手にした実用LISPがこのmuLISPでした．勝手にハワイアンLISPと呼んでいる米国製のLISPです．インタプリタの計算が速いのが特徴で，Common LISPに文法を合わせてあります．コンパイラが付属しています．

本体はコンパクトで，独自のLISPです(厳密に言えば，Common LISPではありません)．



〈写真２〉muLisp

この本体も良いのですが，拡張のための添付のLISPプログラムがまたよくできています．任意精度の三角関数など他では見たことがありません．

オブジェクト指向を語る人には必須の知識である，Flavorsや後述するCLOS(Common LISP Object System)も，ライブラリの一つとしてお試しできます．

LISPで書かれたEMACS風のエディタが付属していて，筆者はしばらくこのエディタを使いたいがために，muLISPをディスク(磁気バルブ)に入れ続けていたことがあります．

また，これらのLISPプログラムのソースが付属してくるのです．お買得と言えるでしょう．

数値データは多倍長整数と，二つの多倍長整数による分数のみです．独立した文字列型はなく，印字名(後述)を文字列として使います．配列は，ライブラリでの対応となっています．

このLISPは，使いこなしたい人向きのLISPだと思います．もちろん，Common LISP系のLISPなので操縦も楽ですから，初心者でも安心して購入できます．

● **フリーソフトウェア**

HOBBES CD-ROMという，OS/2のプログラムを集めたCD-ROMに，二つのコンパクトなLISPが入っています(XLispとXScheme)．

インターネットにアクセスできる方は，ftpサイトを探してみてください．

● **ワークステーション，大型機**

ワークステーションのLISPも広告では少数です．高価ですが，試してみたいLISPの名前が上がっています．

大学の大型計算機には，なぜか優良なLISPが入っていたりします．TSSでLISPと打ってみてください．計算機使用料がちょっと心配になりますが．

基本は対話システム

多くのLISPの教科書の前半は，延々とデータの話が続き，ちっともプログラミングの話が出てきません．それには理由があります．

もともとLISPは，リストなどのLISPの式を与えてその結果を得る，対話システムです．

つまり，BASICふうに書くと，

```
while 1
  input x
   ... '入力された文字列，xを計算し，
       結果をyに代入する
  print y
end while
```

というプログラムが最初から動いており，このルー

プを脱出するとOSに制御が戻ってしまうのです.

ここでいう,「文字列を計算する」は, BASICで説明すると,

```
print a+5
```

の入力が「変数**a**の内容を参照し, **5**を足して」,「その結果を印字する」, という動作に相当します.

LISPの場合, **print**を書かなくても勝手に計算されてしまう, そのような感じです.

例えば, LISPのプロンプト(>とします)で,

```
>(sin 5)
```

と入力すると, **sin(5)**の値が印字されるはずです. 開カッコの位置が, 左に一つずれています. これは, LISPの記法の習慣です.

つまりLISPでは, プログラムを書かなくてもとりあえず計算ができてしまうのです. 対話システムのないLISPはおそらく存在しないでしょう.

● **関数の定義**

関数の定義は, プロンプトからもできます(以下, Common LISPの書法で説明).

```
>(defun fact (n)
   (cond ((<= n 1) 1)
     (t (* n (fact (- n 1)))) ))
```

などと, 関数の定義を直接手で打つことも可能です. プロンプトは, カッコのバランスが取れた, 3行目の入力後に返って来ます.

ちょっと書き方が変わってはいますが, CやBASICの関数定義と本質的には変わりません. Cに書き直すと,

```
int fact (int n) {
  if (n  <= 1) {
    return 1; }
  else {
    return fact (--n) * n; } }
```

などとなります.

condはCの**if .. else if .. else if ..**に相当します. **t**はtrue(真)の略です.

ここで,

```
>(fact 120)
```

などとすると120の階乗が計算され, 表示されます. もちろん, 関数定義はエディタで書くのが普通です. そして,

```
>(load "prog1.lsp")
```

などとロードして, 実行します. 懐かしのBASICと同じです.

● **変数**

上記の階乗のプログラムで, **n**は明らかに局所変数です. しかし, LISPの変数の振る舞いには特別の注意が必要です.

LISPの変数は, Cや通常のBASICのように, 領域に付けられた名前ではありません.

LISPの変数に値を「代入(前の値を破棄して新しい値に置き換える)」する特殊形式(後述)は, **setq**(set quoteの略)です.

```
>(setq a 5)
5
>a
5
```

2回目の入力は**a**のみですが, BASICでの**print a**に相当する操作と考えてください.

```
>(setq a "hello.")
```

```
"hello."
>a
"hello."
```

つまり，変数 **a** には数値だけでなく，文字列も「くっつく」のです．この「くっつき」を，束縛(binding)といいます．

つまり，**A** というデータ(後述しますが，シンボルという型のデータ)に，**5** というデータや **"hello."** というデータが結び付いたのです．

A のような大域変数の場合，シンボル **A** を表す構造体のメンバの一つが値をポイントすることになります(後述)．

前述の **N** のような局所変数の場合，**N** と **120** へのポインタのある内部的な表が(例えばスタック上に)あると考えてください．

● 特殊形式

特殊形式(special form)は，LISP 用語の中でも，初心者泣かせの一つでしょう．特殊形式とは，関数のように書けるが関数でないものです．form は，「式」の意味です．

特殊形式は，**if** とか **defun** とかですから，つまりは，普通の言語での **if...** とか **while...** とか **'...'** とか **{...}** とかを，むりやり関数の形に書いたものだと言えます．

例えば，**if** は特殊形式です．

```
(if 式1 式2 式3)
```

は，**式1** が偽(nil)でなければ**式2** を計算し，結果を返します．**式1** が偽なら**式3** を計算し，その結果を返します．

もしも，**if** が関数なら，**式1**，**式2**，**式3** はすべて計算され，**if** という関数の定義が実行されるはずです．

式2 と**式3** のどちらかは計算されないのですから，形は関数ですが実態は関数でないことになります．

なぜ，このような制御構造を関数の形に似せるかというと，書式を統一してプログラムをデータとしても扱えるようにするためです．

LISP シンボル

シンボル(記号)は，LISP で最も基本となるデータです．BASIC では文字列と数値が基本的なデータです．

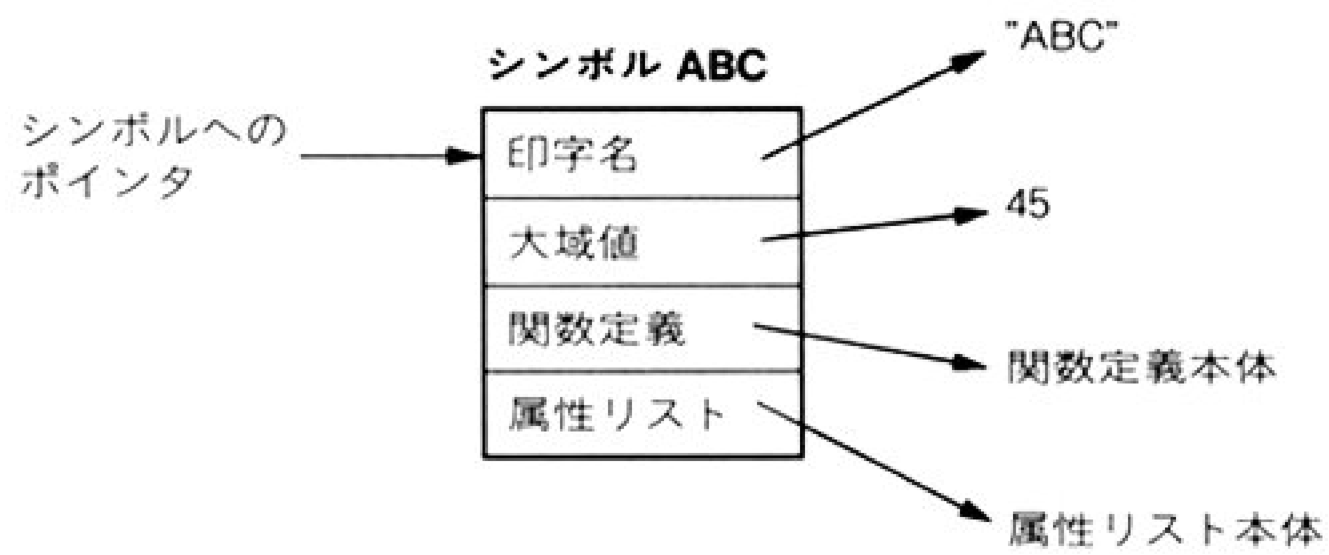
LISP にももちろん数値と文字列はありますから，シンボルは第三の基本データ型と言えます．

シンボルは，実は BASIC にも C にもあります．**if** とか **while** です．「ええっ，これはデータではないよ」という声が聞こえてきそうです．

しかし，BASIC インタプリタや C コンパイラにとってはどうでしょう．**if** も **while** も処理の対象です．

シンボルは，文字列と似ています．しかし違うのは，例えば **while** を **wh** と **ile** に分解した場合，**wh** と **ile** は文字列ですが，もはや **while** の意味はありません．

〈図5〉シンボル構造体(概念図)



この性質のため，かつてシンボルは数値とともにアトム(atom)と呼ばれていました．**atom** は「これ以上分割できないもの」という意味です．

● シンボルの中身

シンボルは，内部的には**図5** のような四つのポインタを持つ構造体です．ただし，これは説明のための例です．処理系によって実現方法は異なるでしょう．

これをみるとわかるように，シンボルのつづり(pname＝print name，印字名，文字列型)は，シンボルというデータのメンバ(LISP 用語では属性)であることがわかります．

シンボル型と文字列型は異なります．シンボルから印字名を得る関数，SYMBOL-NAME があります．

第2のメンバは大域値(へのポインタ)です．プロンプトでシンボルをいきなり入力すると，この大域値が返されます．

大域値が定義されていないと「値がない(unbound)よ！」と怒られます(BASIC と同じ)．ちなみに，数値の値はその数値です．文字列の値はその文字列です．

3番目のメンバは関数定義(へのポインタ)です．上述の **defun** は **FACT** というシンボルと，階乗を計算する関数定義(リストまたは機械語)を束縛していたのでした．

特殊形式と後述するマクロの定義もここに入ります．なお，Scheme では大域値と関数定義は同じポインタを使います．つまり，両者を同時に変数に束縛することはできません．

4番目は属性リスト(へのポインタ)です．属性リストは，名前とその値の対応表です．流儀によっては，印字名も大域値も関数定義も属性リストに押し込めます．

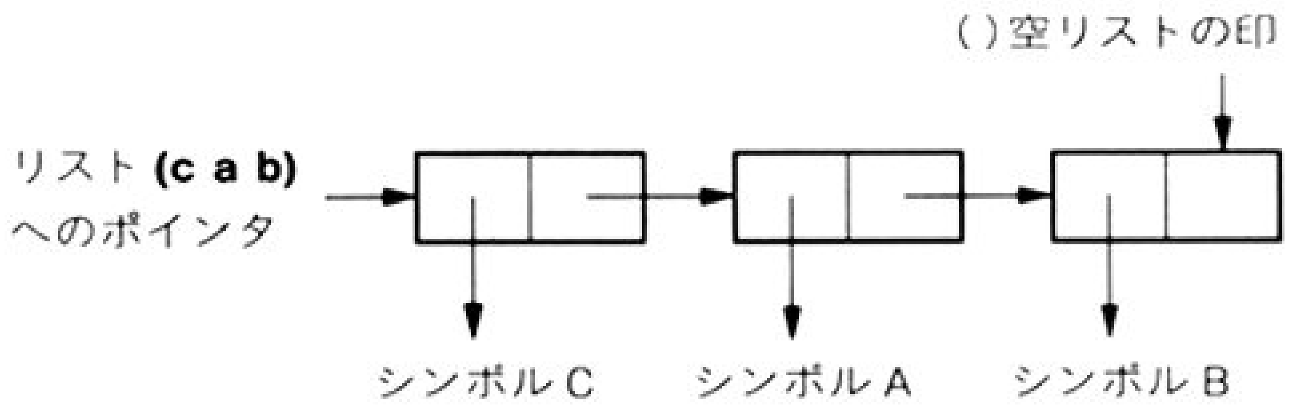
つまり，シンボルの属性を保持するリストです．現代 LISP には，ハッシュ表など，データ同士を結び付けるための便利な仕組みがあるので，かつてほどには使われないようです．

なお，歴史的理由から，LISP ではシンボルとして入力した小文字はすべて大文字に変換されます(Scheme では小文字になる)．

〈図 6〉consとリスト(一例)



(a) cons構造体



(b) リスト(c a b)

変換を阻止するには，エスケープ文字(¥と|)を用います．例えば，印字名が“**abc**”であるシンボルを表したい時には，

|abc|

などとします．

● **シンボルの出現位置**

さきほどから言っている，**A**は大域変数だとか，**N**は局所変数だとか，**SIN**は関数だとか，**IF**は特殊形式だとかは，つまりはシンボルの出現位置による解釈だったわけです．

LISP リスト

リストは，シンボルやリストなど，LISPのデータを空白で区切って，カッコで区切ったものです．かつてはLISPの唯一の複合データでした．

プロンプトに，いきなりリストを書くと，関数として評価〔先に「計算」と述べたのは，LISPでは評価(evaluation)と呼ばれます〕されることはすでに述べました．

評価を避ける，つまりプログラム中にリストを書くには，**quote**という特殊形式を用います．

>(quote (a b c))

と書くと，一瞬リスト**(a b c)**ができます(表示された後，すぐに消滅してしまいますが)．

さすがに毎回**quote**をつづるのはつらいので，ほぼどんなLISPでも，

>'(a b c)

と書けます．**'(a b c)'**の右のアポストロフィを省略した感じです．

● **cons**

プログラムでは，リストは**cons**(コンスと読む)と呼ばれるLISPの関数で作ります．**cons**はconstructの略です．**cons**は，リストの先頭に要素をくっつけたリストを作る関数です．

> (cons 'c '(a b))

(c a b)

この**cons**関数で生成されるデータも**cons**と呼ばれます．**cons**構造体(図6)には，**car**と**cdr**(クダーと読む)の二つのメンバーがあります．即値が入りうるのでややこしいのですが，概念的には**car**と**cdr**はLISPのデータへのポインタと考えればよいでしょう．

carは“Content of Address part of Register”，**cdr**は“Content of Decremental part of Register”の略です．この名称の由来は，IBM 704という計算機の上で最初に作られたLISPシステムによるとのことです．

carと**cdr**はLISPの関数でもあります．**car**は**cons**の**car**部，すなわちリストの先頭要素を返し，**cdr**は**cons**の**cdr**部，つまり先頭要素を取り去った残りのリストを返します．

>(car '(c a b))

c

>(cdr '(c a b))

(a b)

>(cdr '(a b))

(b)

>(cdr '(b))

()

最後の**()**は空リストです．内部的には，NULLポインタなどで表されます．この空リストはアトムであり，かつリストである唯一のデータです．

リストの記法のため，**car**関数と**cdr**関数の効果が対称に見えません．**cons**構造体では**car**と**cdr**は対等ですから，対称的なデータも作れます．しかし，LISPでは上記のような(非対称な)リストを使うことの方が多いのです．

● **基本関数**

consと**car/cdr**，同じデータを指すかどうかを判別する**eq**関数，リストかアトムかを判別する**atom**関数は，LISPの基本関数と呼ばれます．

これに**quote**特殊形式と**cond**特殊形式(Common LISPでは，**cond**はマクロ)があれば，基本的にはリスト操作が自由自在にできます．

これらの関数は，LISPの関数の説明に用いられます．つまり，LISPの習慣として，言葉で説明するよりLISP自身で関数の動きを説明するのです．

ただし，基本関数では**car**部/**cdr**部に直接代入できませんから，新しいリストは必ず**cons**でせっせと作ることになります．

直接代入版は高速ですが，副作用も激しいので，リスト処理がよくわかっていない間は，基本関数から導かれる関数のお世話になることになります．

● **リストの使い方**

要素が数値のリスト，

(1 2 3)

は配列にも見えますし，構造体にも見えます．

Common LISPには，配列も構造体もありますし，みなさんはこのリストが三つの**cons**で構成されることを知っていますから，これは配列でも構造体でもない，と結論されるでしょう．

ところが，もともとのLISPの複合データはリストしかなかったので，かつてリストは，配列的にも構造体的にも用いられました．

配列とリストと構造体の区別は大切なのですが，LISPではたまたま同じリストを使うので，他の言語の学習者にはとまどいの原因になります．ちなみに，同じくリストをよく使うPROLOGでは，構造(複合項)とリストはまったく別の記法になります．

さすがに配列がないとあまりに効率が悪いので，配列は早々にLISPに取り入れられたようです．しかし，配列や構造体よりもリストのほうが柔軟性がありますし，LISPではリスト処理が美しく書けるので，リストはいまでもLISPの最も大切な複合データです．

早い話，関数がリストの形をしています．

(sin 5)

は，**sin(5)**の意味に評価されますが，データとしては，**SIN**というシンボルと**5**という数値を空白で区切って並べ，カッコでくくったリストです．つまり，プログラム自体が，リストで表されています．

● マクロ

プログラムがリストで表されているのなら，プログラムの変換もLISPで可能なはずです．まさにその機能がLISPにはビルトインされています．マクロです．

マクロといえば，Cの**#define**を思い出します．基本的には同じことです．しかし，LISPでは特別なマクロプロセッサがあるのではなく，LISPという強力な記号処理言語がマクロを展開します．そのためLISPのマクロはおそろしく強力です．

● evalとapply

関数がデータなら，データを関数として実行する関数があってもよいはずです．**eval**と**apply**は，その目的のための関数です．

前にBASICの**while**ループを持ち出しましたが，いまやLISPでLISPインタプリタの動作を記述できます．

(defun lisp( ) (print(eval(read))) (lisp))

read関数は，端末から一つのLISPの式(シンボルや数値やリスト)を読み込む関数です．

もちろん，肝心なのは**eval**の定義です．簡略化したLISPシステムの**eval**はLISPで40行ほどになります．この関数は万能関数と呼ばれます．

万能とはおおげさな名前ですが，universalの訳であり，universal jointという言葉があるように，適用範囲が広い，というほどの意味です．

上記のLISPインタプリタにはちょっとごまかしがあり，リストを並べると逐次実行するという約束(たいていのLISPに存在する)を利用しています．

もしきちんとしたければ，**progn**という逐次処理の特殊形式を用いることになりますが，いずれにしろLISPの範囲内でことが済みます．

最近は万能関数をあまりうるさく言いません．教科書に載っているような小さい**eval**の実用価値はほとんどゼロだからです．ただし，LISP処理系の動作の説明には役立ちます．

● 末尾再帰

関数が自分自身を呼び出す動作を再帰(recursion)と言います．

関数を呼び出すには，戻ってから続きを実行するために，その時点の情報をスタックに保存する必要があります．

ですから，何回も再帰を繰り返すとメモリをあっという間に使い果たしてしまいそうです．実際，通常の再帰ではそうなっています．

ところが，たとえば上述のLISPインタプリタでは最後の**lisp**関数を呼び出した時点で，呼び出した方の関数**lisp**に用事は残っていません．単純に値を返すのみです．

このような事態をシステムが発見すると，スタックに情報を保存しないことがあります．要するに，関数コールが**goto**に自動変換されたわけです．

Schemeでは，この末尾再帰(tail recursion)の最適化(**goto**への変換)は必須です(**リスト1**)．

〈リスト1〉再帰(test1.lsp)

```
; test1.lsp YOkada 950326
; 再帰してもスタックがあふれない例。
; 末尾再帰するためには、pf1のような、
; 補助関数を考えると作りやすい。
; use TUTScheme
; (load "test1.lsp")
; (print-forever)
; 止めるには、Ctrl-C。

(define (print-forever)
  (pf1 0) )

(define (pf1 n)
  (display n)
  (display '| |) ;print space
  (pf1 (1+ n)) )
```

LISP 名前の管理表

シンボルを端末から打ち込んだとき，そのシンボルの意味をシステムが知っていなければなりません．

LISPがその時点までに入力したシンボルは，すべて内部的な管理表に登録されています．

もし，この中にないシンボルが入力されたら，表のレコードを一つ増やして，上述のシンボル構造体を登録します．この登録動作をインターン(intern)と言います．

この動作は，OS の作業と同じです．OS のコマンドインタプリタは，何か文字列が入力されたら，適当な応答をしなければなりません．

つまり，OS はシステムに何があるか，それが指定されたら何をすべきかを知っています．同様に，LISP 対話システムは，小さな OS とも言える世界を形成しています．

● read 関数

シンボルがインターンされるのは，主に入力時です(プログラム中の **intern** 関数でインターンすることもできる)．つまり，LISP の **read** 関数は，文字列を読み込むだけでなく，シンボルやカッコを見て，名前の管理表を更新したり，**cons** でリストを作ったりと，かなり複雑なことをする関数です．

この高機能な **read** 関数のおかげで，その後の記号処理が楽になります．

● 入出力

read 関数は複雑ですが，もっと単純な C の **getc** や BASIC の **INKEY$** に相当する関数が LISP にもあります．少なくとも MS-DOS や UNIX の LISP は，入出力はバイトストリームとして扱いますから，他の言語と何ら変わりはありません．

OS/2 などでは，入出力はファイルだけでなく，パイプに接続することも可能です．

ガベージコレクション

setq などを使うと前の値は放棄されるのですから，何もしないとデータが宙に浮いてしまうことになります．

それだけでなく，内部的に **cons** などは一時的にかなり大量に作られているとのことです．

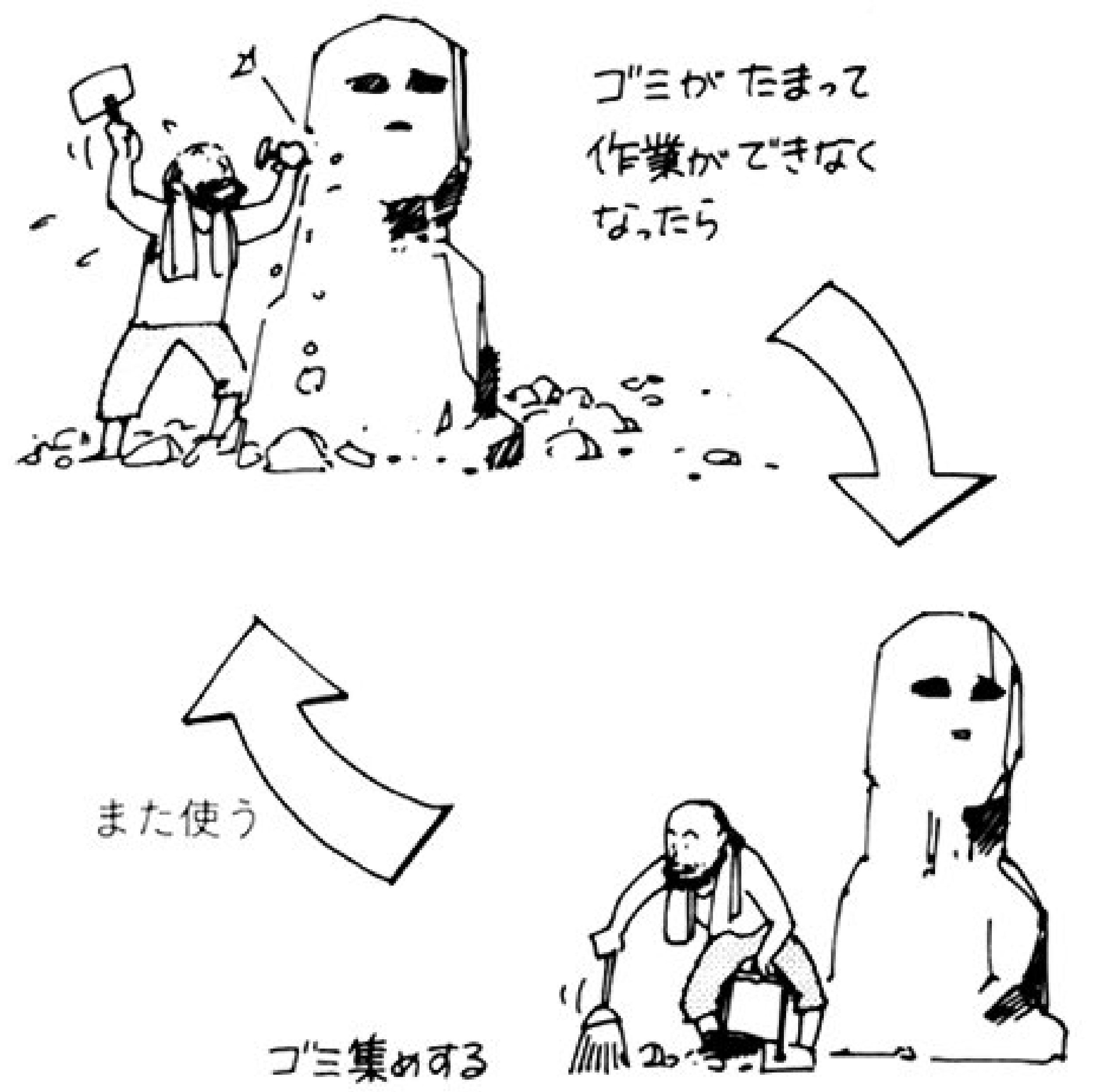
しばらくすると，メモリには使われなくなったデータが大量にゴミとして残ってしまうことになります．

ごく初期の LISP からこのゴミを回収して，新たなデータ領域の確保に使うテクニックが使われていました．ガベージコレクタ(ゴミ集め)です．

C＋＋ならデストラクタに細心の注意が必要，といったことを気にすることなく，ガベージコレクタがあるので，ゴミを発生しながらどんどんプログラミングができます．

代償として，ごみ集めをする CPU 時間が別に必要となります．ただし，その都度の **free** は不要になるので，ガベージコレクションの有無による平均効率は議論の対象になるでしょう．

〈図 7〉ガベージコレクション



LISP でループさせながら印字していると，ときどきしゃっくりのように，コンマ何秒か止まったように見えるのが，ガベージコレクションです(図 7)．

カベージコレクタの研究には古い歴史があり，仮想記憶で効率のよい方法など，いくつものアルゴリズムが開発されています．

CLOS (Common Lisp Object System)

CLOS(シーロス，Common LISP Object System)は，Common LISP のオブジェクト指向への拡張です．

メッセージ伝達の概念がない，メソッド(C のメンバー関数に相当)は動的(実行時)に選択されるのに，効率のよい実装法が存在する，など話題も豊富です．

CLOS 処理系は少数ですが購入可能です．また，書籍もありますので，オブジェクト指向を知りたい人で余裕のある方は調べてみてはいかがでしょうか．

◆ 参考文献 ◆

(1) 竹内郁雄；プロシン版マーフィーの法則，bit 26(12)14-17，共立出版，1994．
(2) G. L. Steele Jr.；COMMON LISP 第 2 版，共立出版，1991．
(3) 竹内郁雄；初めての人のための LISP，サイエンス社，1986．
(4) 湯浅太一；Scheme 入門，岩波書店，1991．
(5) M.Eisenberg；Programming in Scheme，The MIT Press，1990．
(6) 湯浅太一，ほか；Common Lisp 入門，岩波書店，1986．
(7) 稲葉浩人，ほか；Common lisp オブジェクト指向(CLOS)，トッパン，1991．
(8) 井田昌之，ほか；bit 別冊 Common Lisp オブジェクトシステム -CLOS とその周辺-，共立出版，1989．
(9) 坂村健；コンピュータ・アーキテクチャ -電脳建築学-，共立出版，1984．

PC98シリーズ全モデルの変遷

<p.65から>

発売時期	機種	価格(単位千円)	CPU
1976年 9月	TK-80	88	1
1977年 11月	TK-80E	60.7	1
	TK-80BS	128	
1979年 8月	PC-8001	168	1
1981年 7月	N5200 モデル 05	748	2
11月	PC-6001	89.8	1
12月	PC-8801	228	1
1982年 9月	PC-9801	298	2
1983年 1月	PC-8001 mk II	123	1
7月	PC-6001 mk II	84.2	1
10月	PC-9801 F1	328	2
	PC-9801 F2	398	
	PC-100 model 10	398	2
	model 20	448	
	model 30	558	
11月	PC-8801mk II		
	model 10	168	1
	model 20	225	
	model 30	275	
	PC6601	143	1
	PC-9801E	215	2
1984年 10月	PC-9801 F3	758	2
11月	PC-6001 mk II SR	89.8	1
	PC-6601 SR	155	1
	PC-9801 M2	415	2
1985年 1月	PC-8001 mk II SR	108	1
	PC-8801 mk II SR		
	model 10	168	1
	model 20	213	
	model 30	258	
2月	PC-9801 M3	838	2
5月	PC-98XA		
	model 1	575	4
	model 2	695	
	model 3	1,035	
6月	PC-9801 U2	298	3
7月	PC-9801 VF2	348	3
	PC-9801 VM0	295	
	PC-9801 VM2	415	
9月	PC-8801 mk II TR	288	1
10月	PC-9801 VM4	830	3
11月	PC-8801 mk II FR		
	model 10	99.8	1
	model 20	148	
	model 30	178	
	PC-8801 mk II MR	238	1
1986年 5月	PC-98XA		
	model 11	495	4
	model 21	575	
	model 31	865	
6月	PC-9801 UV2	318	3
10月	PC-98LT model 2	288	11
	PC-9801 VX2	433	3, 4
	VX4	693	
11月	PC-9801 VX0	353	3, 4
	PC-98LT model 1	238	11
	PC-9801 VM21	390	3
	PC-8801 MH	208	1
	PC-8801 FH		
	model 10	99.8	1
	model 20	138	
	model 30	168	
12月	PC-9801 VX4/WN	730	3, 4
	PC-98XL model 1	495	3, 4
	model 2	575	
	model 4	835	
1987年 2月	PC-8801 FH model 30 ブラックタイプ	168	1
3月	PC-88VA	298	0

発売時期	機種	価格(単位千円)	CPU
6月	PC-9801 UV21	318	3
	PC-9801 VX01	353	3, 4
	VX21	433	
	VX41	630	
8月	PC-9801 VX41/WN	658	3, 4
10月	PC-98XL^2	988	3, 5
	PC-8801 FA	168	1
	MA	198	
	PC-9801 UX21	348	3, 4
	UX41	545	
	PC-98LT model 11	238	11
11月	PC-98LT model 21	288	11
1988年 3月	PC-9801 UV11	265	3
	PC-9801 LV21	345	3
	PC-88 VA2	298	0
4月	PC-9801 CV21	355	3
8月	PC-9801 RA2	498	3, 5
	RA5	736	
	PC-88 VA3	398	0
9月	PC-9801 RX2	398	3, 4
	RX4	566	
10月	PC-8801 MA2	168	1
	FE	129	
11月	PC-9801 LS2	628	3, 6
	LS5	866	
	PC-9801 VM11	328	3
1989年 2月	PC-9801 LV22	378	3
	PC-98RL model 2	735	3, 5
	model 5	970	
4月	PC-9801 EX2	348	3, 4
	EX4	488	
	PC-9801 LX2	448	3, 4
	LX4	588	
	LX5	638	
	PC-9801 ES2	448	3, 6
	ES5	638	
6月	PC-98DO	298	3
8月	PC-9801 LX5C	748	3, 4
10月	PC-8801 FE2	119	1
	PC-9801 RX21	338	3, 4
	RX51	508	
11月	PC-8801 MC		
	model 1	169	1
	model 2	199	
	PC-9801 RA21	498	3, 5
	RA51	668	
	PC-9801 RS21	398	3, 6
	RS51	568	
	PC-9801N	248	3
1990年 1月	PC-H98		
	model 70-002	995	5
	model 70-100	1,680	
2月	PC-9801 T		
	model W2	548	3, 6
	model W5	758	
6月	PC-9801NS	298	6
	PC-9801NS-20	448	
	PC-9801T		
	model F5	1,150	3, 6
7月	PC-9801T		
	model S5	850	3, 6
9月	PC-98RL		
	model 21	700	3, 5
	model 51	860	
	PC-H98		
	model 60-002	835	5
	model U60-002	835	
	model 60-040	995	
	model U60-040	995	

発売時期	機種	価格(単位千円)	CPU
	model 60-100	1,500	5
	model U60-100	1,500	
	PC-H98		
	model 100-100	2,150	7
	model U100-100	2,150	
10月	PC-98HA	198	11
	PC-98DO$^+$	278	12
11月	PC-9801 NV	248	13
	PC-9801 DX2	318	4
	PC-9801 DX/U2	318	
	PC-9801 DX5	468	
	PC-9801 DX/U5	468	
1991年 1月	PC-9801 DS2	358	6
	PC-9801 DS/U2	358	
	PC-9801 DA2	448	5
	PC-9801 DA/U2	448	
	PC-9801 DS5	508	6
	PC-9801 DS/U5	508	
	PC-9801 DA5	598	5
	PC-9801 DA/U5	598	
2月	PC-9801 UF	218	13
	PC-9801 UR	218	13
	PC-9801 UR/20	348	
	PC-9801 DA7	698	5
	PC-9801 DA/U7	698	
4月	PC-9801 NV(B)	278	13
	PC-9801 NV(P)	278	
5月	PC-H98S		
	model 8-002	548	8
	model U8-002	548	
	model 8-040	698	
	model U8-040	698	
	model 8-100	798	
	model U8-100	798	
	PC-9801 NS/E	278	6
	PC-9801 NS/E20	378	
	PC-9801 NS/E40	418	
7月	PC-9801T		
	model W7	750	3, 6
	model F51	850	
	model F71	950	
10月	PC-9801 CS2	398	6
	PC-9801 CS5	548	
	PC-9801 NC	598	6
	PC-9801 NC40	738	
11月	PC-9801 CS5/W	675	6
	PC-98GS model 1	698	3, 6
	model 2	828	
1992年 1月	PC-9801 FA2	458	8
	PC-9801 FA/U2	458	
	PC-9801 FA5	578	
	PC-9801 FA/U5	578	
	PC-9801 FA7	648	
	PC-9801 FA/U7	648	
	PC-9801 NS/T	288	14
	PC-9801 NS/T40	398	
	PC-9801 NS/T80	458	
	PC-9801 NL	238	13
5月	PC-9801 FS2	348	6
	PC-9801 FS/U2	348	
	PC-9801 FS5	468	
	PC-9801 FS/U5	468	
	PC-9801 FS7	538	
	PC-9801 FS/U7	538	
	PC-9801 FX2	278	6
	PC-9801 FX/U2	278	
	PC-9801 FX5	398	
	PC-9801 FX/U5	398	
7月	PC-9801 US	248	6
	PC-9801 US40	358	
	PC-9801 US80	418	
	PC-9801 NS/L	288	6
	PC-9801 NS/L40	398	

0：μPD9002(V30/Z80命令コンパチブル)

PC98シリーズ全モデルの変遷

発売時期	機　種	価　格 (単位千円)	CPU
10月	PC-9821		
	model S1	318	6
	model S2	438	
	PC-9801 NA/C	498	8
	PC-9801 NA/40C	588	
	PC-9801 NA/120C	638	
	PC-9801 NA	348	
	PC-9801 NA40	438	
	PC-9801 NA120	488	
	PC-H98T		
	model 1	820	8
	model 2	980	
	model 2C	1,750	
	PC-H98		
	model 105-100	980	9
	model 105-300	1,180	
	model U105-100	980	
	model U105-300	1,180	
	SV-H98model 41	2,300	
	model 61	5,900	
1993年 1月	PC-9801 BX/U2	218	8
	PC-9801 BX/U6	288	
	PC-9801 BX/M2	232	
	PC-9801 BA/U2	328	9
	PC-9801 BA/U6	388	
	PC-9801 BA/M2	342	
	PC-9821 Ae/U2	358	8
	PC-9821 Ae/U7	438	
	PC-9821 Ae/M2	372	
	PC-9821 Ae/M7	445	
	PC-9821 As/U2	448	7
	PC-9821 As/U7	498	
	PC-9821 As/U8	580	
	PC-9821 As/M2	462	
	PC-9821 As/M7	505	
	PC-9821 Ap/U2	550	9
	PC-9821 Ap/U7	600	
	PC-9821 Ap/U9	830	
	PC-9821 Ap/M2	564	
	PC-9821 Ap/M7	607	
	PC-9801 NS/R	288	8(1)
	PC-9801 NS/R40	378	
	PC-9801 NS/R120	428	
5月	PC-9821 Ce		
	モデル S1	378	8
	モデル S2	488	
	PC-9821 Ae/U7W	488	8
	PC-9821 Ae/M7W	495	
	PC-9821 As/U7W	550	7
	PC-9821 As/U8W	630	
	PC-9821 As/M7W	557	
	PC-9821 Ap/U7W	650	9
	PC-9821 Ap/U9W	880	
	PC-9821 Ap/M7W	657	
7月	PC-9821 Ne	588	8
	PC-9821 Ne120/W	728	
	PC-9801 NX/C	438	8(1)
	PC-9801 NX/C120	548	
	PC-9821 Af/U9W	1,200	15
	PC-9821 Af/M9W	1,200	
	PC-9801 P40/D	498	8(1)
	PC-9801 P80/W	560	
	PC-9801 P80/P	560	
11月	PC-9821 Be/U7W	268	8
	PC-9821 Bs/U7W	298	7
	PC-9821 Bp/U7W	398	9
	PC-9821 Bp/U8W	438	
	PC-9821 As2/U2	328	8
	PC-9821 As2/M7	342	
	PC-9821 As2/U7W	418	
	PC-9821 As2/U8W	458	
	PC-9821 Ap2/U2	448	9
	PC-9821 Ap2/M2	462	
	PC-9821 Ap2/U8W	580	
	PC-9821 Ap2/C9W	700	
	PC-9821 Ap2/C9T	800	
	PC-9821 As2/U7P	418	8
	PC-9821 Ap2/U8P	580	9
	PC-9801 BX2/U2	178	8
	PC-9801 BX2/M2	192	
	PC-9801 BX2/U7	238	
	PC-9801 BS2/U2	238	8
	PC-9801 BS2/M2	252	
	PC-9801 BS2/U7	278	
	PC-9801 BA2/U2	338	9
	PC-9801 BA2/M2	352	
	PC-9801 BA2/U7	378	
	PC-9821 Ts	670	8
	PC-9821 Ts120/W	770	
	PC-9821 Ne340/W	768	8
1994年 1月	SV-98 モデル 1	540	15
		(旧:750)	
	PC-9821 Bf/U8W	600	15
	PC-9821 Ce2		
	モデル S1	298	8
	モデル S2	348	
	モデル S2D	363	
	PC-9821 Cs2		
	モデル S2	428	8
	モデル S3	468	
	PC-9801		
	NL/R120A	328	13
	PC-9801 NS/A	248	8
	PC-9801 NS/A120	328	
	PC-9801 NS/A340	398	
5月	PC-9821An/U8W	610	15
		(旧:720)	
	PC-9821 An/C9T	700	
		(旧:900)	
	PC-9821 An/U2	500	
		(旧:590)	
	PC-9821 An/M2	514	
		(旧:604)	
	PC-9821 An/U8P	610	
		(旧:720)	
	PC-9821 Ap2/U8P	580	9
	PC-9821 As2/U8P	458	8
	PC-9821 As2/U7P	418	
	PC-9821	640	8(1)
	Np/340W	(旧:840)	
	PC-9821	690	
	Np/540W	(旧:890)	
	SV-98	860	
	model 1N1	(旧:1,060)	
	model 1N2	840	
	SV-98 model 2F	720	
	SV-98 model 2	980	15
		(旧:1,100)	
	SV-98 model 2N1R	1,430	
	(SV-98 model 2N1)	(旧:1,560)	
	SV-98 model 2N2	1,720	
7月	PC-9821 Ce/T2	358	8
	PC-9821 Ce/T2D	373	
	PC-9821	740	8(1)
	Np/810W	(旧:940)	
	PC-9821	540	9
	Ns/340W	(旧:730)	
	PC-9821	590	
	Ns/540W	(旧:780)	
	PC-9821	640	
	Ns/810W	(旧:830)	
	PC-9821 Ne2	458	8
	PC-9821 Ne2/340W	578	
	PC-9821 Nd	348	8
	PC-9821 Nd/340W	468	
	PC-9821	840	15
	Xt/C10W	(旧:1,000)	
	PC-9821	520	15
	Xa/U8W	(旧:640)	
	PC-9821	560	
	Xa/C9W	(旧:730)	
	PC-9821	710	
	Xa/C10W	(旧:880)	
	PC-99821 Xa/U1	440	
		(旧:500)	
	PC-9821 Xs/U7W	348	9
	PC-9821 Xs/C8W	398	
	PC-9821 Xs/U7WL	380	
	PC-9821 Xp/U8W	448	10
	PC-9821 Xp/C8W	488	
	PC-9821	448	15
	Xn/U8W	(旧:570)	
	PC-9821	498	
	Xn/C9W	(旧:660)	
	PC-9821 Xe/U7W	238	8
	PC-9821		
	Es/340N	720	9
	Es/540N	770	
	Es/340U	468	
	Es/340NL	760	
	Es/340NP	720	
	Es/340UP	468	
10月	PC-9821 Cb		
	モデル 2F	355	8
	モデル 2D	340	
	モデル 2	325	
	PC-9821 Cx		
	モデル S2	415	8
	モデル S3	455	
	PC-9821		
	Cf モデル S3	590	15
	PC-9821 Ap		
	3/C8W	560	10
	3/C9W	600	
	3/U2	448	
	3/M2	462	
	PC-9821 As		
	3/C8W	458	9
	3/U2	349	
	3/M2	362	
	PC-9821 Ld/350A	398	8
	PC-9821 Ld/260	348	
	PC-9801 NL/A260A	288	
	PC-99801 NL/A120	248	
11月	PC-9821 Xf/C9W	498	15
	PC-9821 Xf/U8W	448	
	PC-9821 Xf/U1	370	
12月	SV-98		
	モデル 1N3	730	15
	モデル 2N3	1,360	
1995年 1月	PC-9801		
	BA3/U2	138	9
	BA3/U2/W	198	
	PC-9801		
	BX3/U2	98	8
	BX3/U2/W	163	
	PC-9821		
	Nf/340W	690	15
	Nf/810W	790	
	PC-9821		
	Lt/540A	488	9
	Lt/340A	438	
	Lt/260	388	
	PC-9821 Nm	188	9
	PC-9821 Nm/340	248	

1：8080またはZ80/互換CPU　4：80286　7：486DX　10：DX4　13：V30HL　(1) 16ビットバス版
2：8086　5：80386　8：486SX　11：V50　14：386SL
3：V30　6：80386SX　9：DX2　12：V33A　15：Pentium

新連載

パソコン草創期ソフトウェア回顧録

第1回　パソコン言語との出会い

西田雅昭

筆者は子供の頃からの秋葉原少年で，TTLでマシン制御のシステムを組むといったことを趣味半分でやっていましたが，文系出身で情報系の教育を受けたことはありません．

しかし，好きなこともあって草創期からパソコンにつき合うことになってしまいました．そこで，筆者がこれまでつき合ってきたソフトウェアを振り返りながら日本のパソコンソフトの歴史を紹介したいと思います．ただし，すべてのソフトウェアを使いこなしたわけではありませんので，昔話風の印象批評になってしまうことをお許しいただきたいと思います．

パソコンの言語との出会い

筆者のパソコン用言語との出会いの中で，最も印象に残っているのは，Tandy DOS(注1)の上で走った「Pascal 80」でした．これは，**with**がないなど，制約の多いPascalでしたが，筆者の言語に対する考え方を根本から変えたものでした．もっとも，インパクトは言語そのものにあるのではなくて，Niklaus Wirthが書いた「PASCAL」(注2)という本が，言語の論理性や美しさを教えてくれたせいかもしれません．

● TRS-80の時代

それまでの筆者は，Tandy BASIC(注3)上でプログラムを短く書くことを誇っていたもので，1行に多くのステートメントを書き，リストを取って真っ黒な長方形になるのを恥だとは思っていませんでした．トリッキーの上にトリッキーを重ねるという世界です．

TRS-80というパソコンには多くのDOSが存在し，そのいずれもシステムを自由にいじることができ，プログラム開発用として当時としては優れた環境を持っていました．



〈写真1〉
Jensen & Wirth著
「PASCAL」

また，素晴らしいモニタソフトがあり，DISASSEMBLERもDEBUGGERも揃っていました．例えば，「NEWDOS80」というDOSなどは，今でもMS-DOSのバージョン2よりも優れたものだと思っています．あの頃は，いくつものDOSを買いあさって，言わばOSオタクでした．そのせいか，アメリカからのサポートも素晴らしいものを享受しました．「NEWDOS80」については，数年間にわたって膨大な量のバージョンアップを無料で受け取っていました．

2ヶ月に一度くらいに，海を越えて送られてくるレポートで，筆者の「NEWDOS80」のマニュアルは，3倍くらいにふくれ上がりました．システムを直接さわることができたので，印刷したアセンブルリストで，バージョンアップができたのです．

筆者だけでなく，Tandyのユーザーはアメリカから資料や雑誌，書籍，ユーティリティソフトなどを入手するのに必死でした．現在でも何かというと，筆者が海外に情報やソフトを求めるのは，この時代から始まっているといってよいかもしれません．また，ショップでの客同士の情報交換も盛んで，ここでできた友人もたくさんいます．

アメリカからは，そのほかにもOSやBIOSに関するたくさんの資料を手に入れることができました．それも，ソースリストだけでなく，回路や石の解説まで付いていました．

ゲームのプロテクト破りもよくやりました．言い訳ではありませんが，ソフトのコピーをしたかったわけではなく(自分で購入したソフトと取っ組むのだから)，知恵比べが面白かったのです．

また，その頃は企業の経営コンサルタントのような

(注1) 当時，モニタ付きのパソコン(マイコンと呼んでいた)の3大勢力は，

TRS-80(Z80)…Tandy RadioShack
AppleII(6502)…Apple
PET(6502)…Commdore

であった．筆者は，ディスクの拡張性を考えてTRS-80を選択した．最初は，メモリ4Kバイトで全角英字しか表示できなかったが，拡張して64Kバイト片仮名まで表示できるものにした．

(注2) Kathleen Jensen & Niklaus Wirth，原田賢一訳：Pascal，培風館．
Pascalの言語仕様と考え方を解説した基本書籍であった．

(注3) 当時のパソコンに載っている言語は，ほとんどがMicrosoftのBASICインタプリタだったが，機種によってすべてシンタックスが違っていた．筆者は，アメリカから拡張版のDOSを大量に購入していたので，拡張されたBASICを使い，胸の内では圧倒的に豊富な機能を誇っていた．

こともやっていたので，このZ80のマシンで，企業の事務処理のためのソフトも作っていました．パソコンOAの走りといえるかもしれません．

いろいろなBASIC

初期のパソコンと言えばBASICでしょう．まずはBASICの変遷についてお話しましょう．

パソコンの最初のBASICというと，ワンボードマイコンの上に載る4K-BASICです．アセンブラのソースリストを見ながら，HEXコードをテンキー(16進)で，打ち込んだ記憶があります．ASCII誌の第2号には，ソノシートでソースコードが提供され，それをテープに移して入力できたので，とても嬉しかったことを覚えています．

ワンボードマイコンの時代には，多くのCPUが存在しました．モトローラの6800，インテルの8080，8085，ナショナルセミコンダクターのSCAMP，東芝のTLCS12などがありました．RCAのCOSMACは，内部レジスタがスタティックだったので，クロック4Hzを作って，「動作が目に見えるだろ」と人に見せて威張っていました．この頃はアセンブラの名手が存在し，HEXコードを恐るべき速さで，プログラムを16進キーに直接書き下ろす猛者もいました．あのミクロサイズのBASICで「スタートレック」を書いていたのですから，当時のプログラミングの世界もなかなかのものであったと思います．

NECのTK-80の上にもう一枚ボードを載せて，キーボードが付いてモニタTVが使えるようにしたTK-80BSが出たときには，大変な騒ぎでした．

この上でのBASICは記号が使えるので，マニュアルに掲載されていた碁のシュミレーションを動かして，コンピュータ上で囲碁対局も行いました．このBASICを使って，雑誌などでいろいろなプログラムが発表されていましたが，数値の比較の際に正負の区別をしない，という重大なバグがありました．メーカーは隠していましたが，マニアの間では公然の秘密でした．

その後，TandyのTRS-80，コモドールのPET，それにAppleIIの3種類のパソコンが競い合う時代がやってきました．この3種類のマシンのBASICは，お互いに内部構造が違うだけでなく，BASICのシンタックスもまったく違うものでした．

画像に強いAppleIIは，この頃からすでに独自の世界を持っていましたが，マイコンのキャデラックと言われるほど高価なマシンでした．とはいっても，初期の頃は外部記憶装置には300ボーのカセットテープしかなかったので，ゲームソフトなどもテープで売られていました．筆者が最初に購入した90Kバイトもないディスクは，パソコン本体よりも高かったように記

〈写真2〉
バインダになっている
NEWDOS 80のマニュアル



憶しています．

国産では，NECのPC-8001とシャープのMZ-80，富士通のFM-8が競っていましたが，これらのマシンに載っているBASICも互換性がありませんでした．

この時代のBASICは，ROM上に載っているか，テープで読み込むタイプでしたが，インタプリタであり，手軽にセミグラフィックスが扱えたので，爆発的な人気となり，「パソコン＝BASIC」といってもよいような状況になってきました．

その後，各社が競って5インチのフロッピディスクを出すに至って言語も本格的になってきましたが，構造的言語というにはほど遠いものでした．

ほとんどが力まかせのプログラミングで，**GOTO**文の山を築き，小さいサイズの中にいかにして多くの内容を叩き込めるかが，プログラマの腕でした．BASIC自体の互換性のなかったこともあり，スピードや機能の追及のために，やたらにハードウェアをアクセスすることになってしまいました．

PC-8801とFM-8の出現で，パソコンは実用機としての機能を，一応は満足するようになったといえます．NECのN_{80}-BASICからN_{88}-BASICへの移行によって，ファイルの入出力，**while**文の導入による制御構造の整備など，言語としての必要条件がそろい，より実用的なプログラムが作れるようになりました．

しかし，プログラミングする側では，旧来のプログラミングの考え方が根強く残っており，なかなか構造化プログラミングの考え方が浸透しませんでした．

パソコンに搭載されたBASICでも，ISAMに近いファイル操作のできる東芝PASOPIAのT-BASIC，リカーシブコールのできるSONYのパソコンなど面白いものもありましたが，あまり評判にはなりませんでした．

パワーのないマシンにとっては，コンパイラが存在しなかったことが最大の問題でした．後になって，MS-DOSの時代にN_{88}-BASIC用のコンパイラが発売されましたが，ごく一部のパッケージソフトに使われただけで，時すでに遅しということになります．

（つづく）

Paradox for Windows ObjectPAL 簡単入門！



音声/画像付きデータベースと検索プログラムの作成

船江明彦

各社から出ているWindows対応のデータベースには，音声/画像を容易に取り扱えるものが多くあります．この点だけでもMS-DOSのデータベースに比べて格段に高機能化されています．

さて，数あるデータベースの中でも，ここに取り上げるParadox for Windows(パラドックス フォーウィンドウズ，以下Paradox)とそのプログラミング言語であるObjectPALを使用すると，簡単に音声画像付きデータベースが作成できます．

そこで，本稿ではParadoxの入門をかねて，ObjectPALを使用したアプリケーションの開発方法がどのようなものであるかを簡単に解説します．

● 作成したプログラム

図1が，今回作成した音声/画像データとキーワードが登録されているデータベース(テーブル)の検索プログラムです．

画面のボタンから，リストに線が引いてありますが，このリストはボタンが押された時に実行される部分などです．

はじめに，このプログラムの動作を簡単に説明しておきましょう．

① **図1(a)**の検索画面の検索データの項目に，検索する文字列を入力する

② 入力した文字列がデータ内にあれば検索一致とし，**図1(b)**の一覧画面を表示する

③ 一覧画面では，一致した複数のデータを表示する

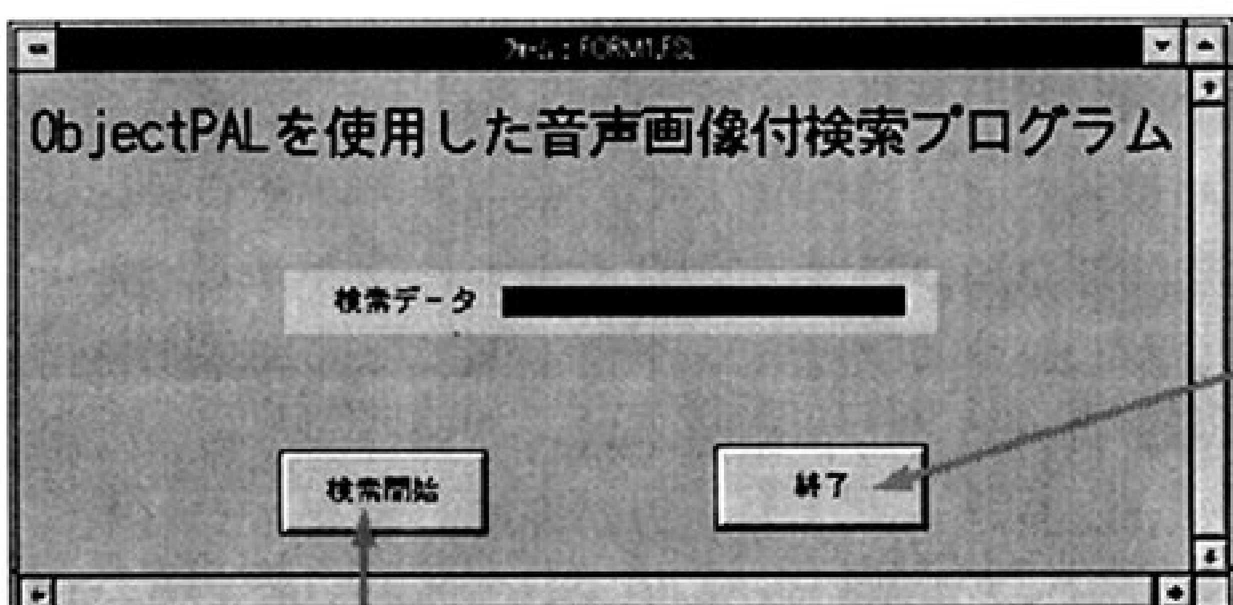
④ [スライド開始]で，検索データを先頭から順に表示し，同時に各データに登録されている音声データも自動的に鳴らす

⑤ [詳細]で**図1(c)**の詳細画面を表示する

このようなプログラムを作成するには，Paradoxに含まれている機能を知っておくだけではなく，その関連性も把握しておかなければなりません．

そこで，プログラムの具体的な作成に入る前にParadoxの提供している機能をどのように考えて使い分けていくかを解説します．

〈図1(a)〉制作した音声/画像データベースの検索プログラムの検索画面(FORM1)




```
method pushButton(var eventInfo Event)
    CLOSE()
endmethod
```

```
method pushButton(var eventInfo Event)
  var
    st          String     ;文字型     変数宣言
    myQbe       Query      ;問合せ型   変数宣言
    answerTbl   Table      ;テーブル型 変数宣言
    formw       Form       ;フォーム型 変数宣言
    i           LongInt    ;整数
  endVar
  st=項目検索データ.VALUE  ;項目の内容を取り込む
  if st="" then
    message ("検索データ に値がありません")
    return
  endif
  st = ".."+st+".."          ;部分一致用
  myQbe = Query              ;問合せ文定義

    データ.db   | 検索データ |
    Check       | Check ~st  |
  endQuery
  message("検索中。しばらくお待ちください")
  if not myQbe.executeQBE() then     ;問合せ実行
    MsgInfo("エラー","問い合わせできませんでした")
    return
  endif
  answerTbl.attach(":priv:ANSWER.DB");問合せ結果のテーブルと結合
  i = answerTbl.nRecords()            ;レコード数 GET
  answerTbl.unAttach()                ;テーブルとの結合解除
  if i =0 then
    message ("検索一致データ がありません")
    return
  endif
  if not formw.open("form2") then
    MsgInfo("エラー","form2が見あたりません")
    return
  endif
  formw.wait()     ;制御をフォームに移す．form2実行
  formw.close()
endmethod
```

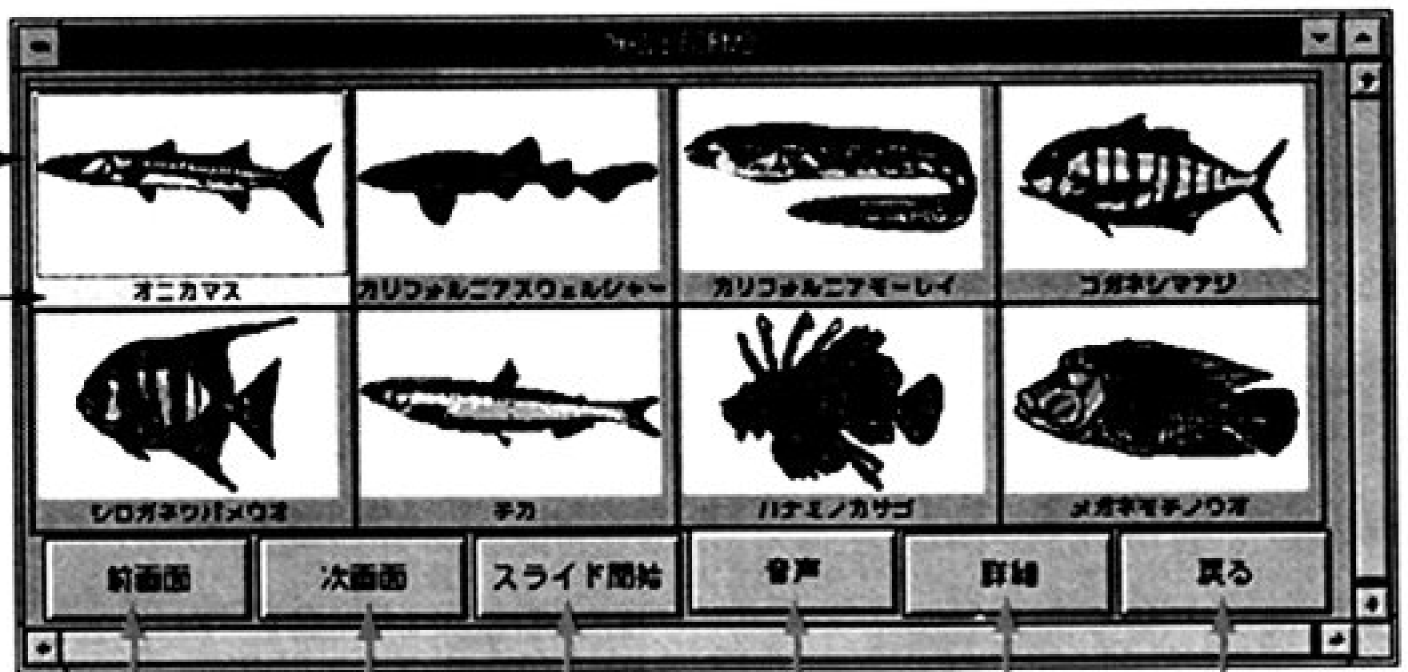
Paradoxの機能概要とその使い分け

● 対話形式の操作と ObjectPAL の使い分け

はじめに紹介した検索プログラムを実際に作成すると、データの登録機能や登録データを維持するためにリスト出力などの機能が必要になります．これらの機能は，プログラミングを必要としない Paradox の対話形式の操作で作成します．

また，このような検索プログラムでは一般利用者にとってわかりやすい操作にすることが必要です．そのためには，ObjectPAL を使用してプログラムを記述し

〈図 1(b)〉 制作した音声/画像データベースの検索プログラムの一覧画面(FORM2)




```
method pushButton(var eventInfo Event)
  formReturn()
endmethod
```

```
method arrive(var eventInfo MoveEvent)
  self.color = Yellow ;レコードを選択色に
endmethod
method depart(var eventInfo MoveEvent)
  self.color = LightBlue ;レコードを非選択色に
endmethod
```

```
method arrive(var eventInfo MoveEvent)
  画像データ.moveto()
endmethod
```

```
method pushButton(var eventInfo Event)
  action(DataPriorSet) ; 前
endmethod
```

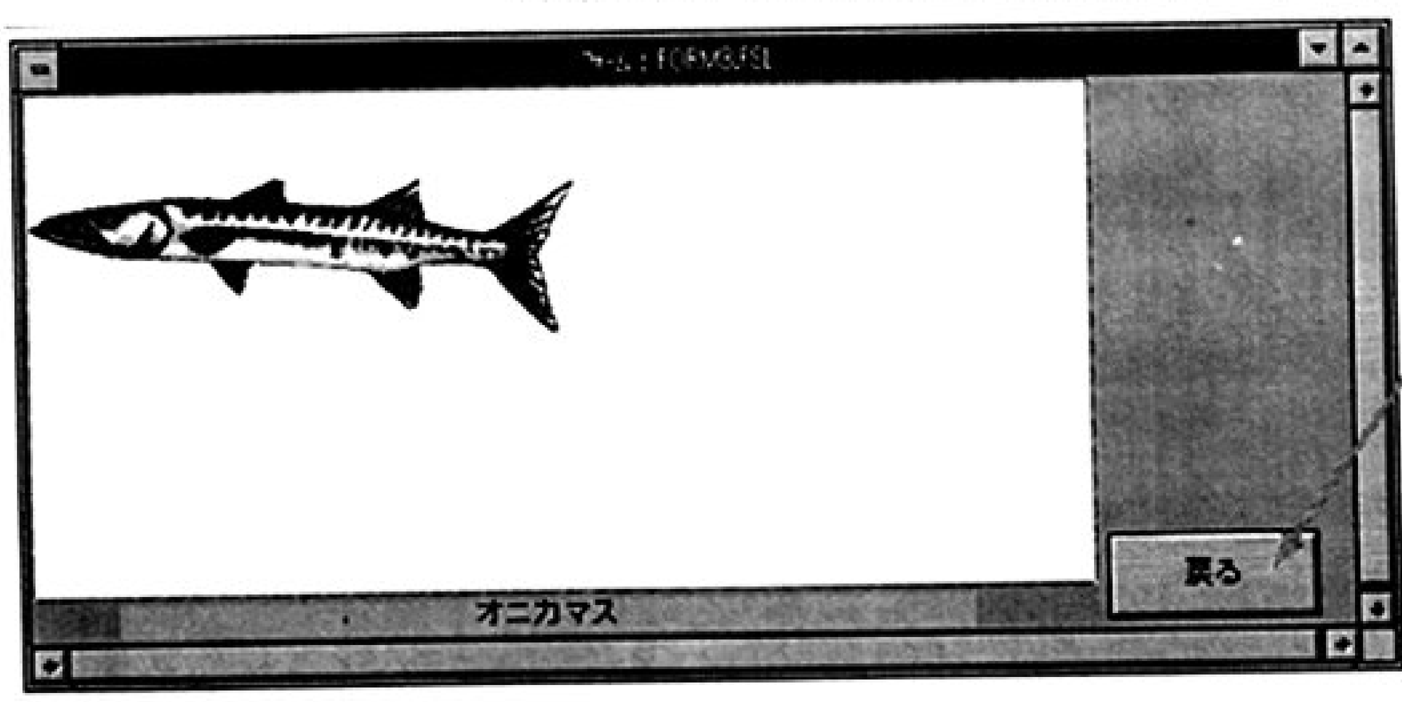
```
method pushButton(var eventInfo Event)
  action(DataNextSet) ; 次
endmethod
```

```
method pushButton(var eventInfo Event)
var
  formw Form
endvar
  if not formw.open("form3") then
    MsgInfo("エラー","form3が見あたりません")
  else
    formw.画像データ.MoveToRecord(検索データ.recno)
    formw.wait() ; form3実行
    formw.close()
  endif
endmethod
```

```
method pushButton(var eventInfo Event)
  var
    i     SmallInt
  endvar
  action(DataBegin)
  message("スライド中")
  for i  ;for はカウンターが必要。ダミー
    #ﾎﾞﾀﾝ18.pushButton( eventInfo );音声ボタン押した処理呼び出し
    if atLast()  then           ;最終レコード
      quitloop
    endif
    action(DataNextRecord)     ;次レコード
  endfor
  message("スライド終了")
endmethod
```

```
var
    anOLE OLE          ;メソッド外側で OLE変数を定義
endvar
method pushButton(var eventInfo Event)
  anOLE = 音声データ
  anOLE.edit("",0)      ;音声起動開始後すぐ戻る
  sleep(3000)           ;3秒 wait.音声データ長さは3秒以内
endmethod
```

〈図 1(c)〉 制作した音声/画像データベースの検索プログラムの詳細画面(FORM3)




```
method pushButton(var eventInfo Event)
  formReturn()
endmethod
```

ていくことになります．

● **管理業務でもObjectPALを活用する**

管理者の業務を対話形式で行う際も，繰り返し入力するパターンや面倒な操作は，やはりObjectPALでプログラムすると効果的です．

従来の言語ではプログラムの中心となる処理から作成しました．具体的には，画面の表示→入力待ち→データの入力に至って初めて処理ができたのです．

ObjectPALでは，ボタンを配置して，そのボタンを押したときにどのような動作をするかを記述できます．そのため対話形式の操作をしながら簡単に処理を組み込むことができます．

● **データ項目の追加と注意点**

データベース(テーブル)にデータを登録して運用する段になると，簡単にデータ項目を追加できるかどうかが大切になってきます．

Paradoxでは，簡単にデータ項目を追加できます．例えばこのプログラムでは，テーブルにデータ項目を追加しても既に動作している画面には影響を与えずに動作します．

読者のみなさんがテーブルに項目を追加しても順次画面を修正して，必要な項目を表示していくことができ，動作中にシステムの変更，拡張が手軽にできます．本格的なシステムでも，サブシステムごとに順次稼働させながら追加し作成することができます．

しかし，動作中に変更を施しテーブルの構造を変えた場合，過去にバックアップしたデータはロードできなくなるので，バックアップ方法を工夫する必要があります．

● **テーブル/フォームとはなにか**

Paradoxでは，データを保存する場所をテーブルといい，テーブルのデータを表現したものをフォームといいます．

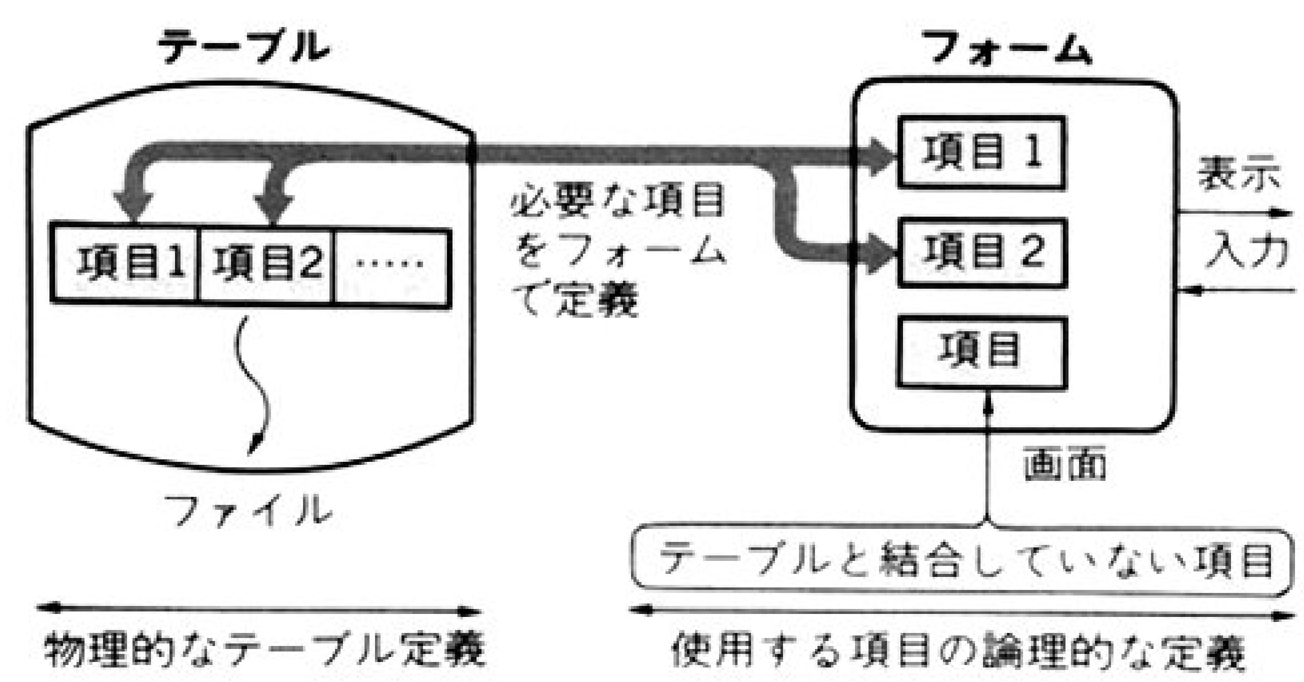
フォームは画面のレイアウトを定義して，テーブルの項目を表示するものです．テーブルとは独立して，データの入出力用の画面を定義できるので，テーブルに項目を追加してもフォームの画面に影響はありません．

図2の概念図のように，フォームはテーブルのデータと人間の間に位置するユーザーインターフェース部分とも言えます．

フォーム，ボタンにはそれぞれメソッドと呼ばれるプログラムを配置できます．特にボタンには押したときに起動するメソッドが定義されています．

さて，Paradoxでは，構成要素となりうるものをすべてオブジェクトと呼びます．具体的には，フォームやテーブルなどを言い，特にフォームに配置するテーブルから項目，ボタンなどをまとめてUIObject(UIはUser Interfaceの略)と呼びます．

〈図2〉テーブルとフォームの概念図



Paradoxでは今回のプログラムの検索画面のように，テーブルの項目をまったく表示しないフォームも作成できるので，データベースと関係のないアプリケーションも作成できると思われます．また，画面でなく印刷のレイアウトを行うものにレポートがありますが，これもオブジェクトです．

● **組み込みメソッドとユーザー処理のアタッチ**

前述のように，ボタンには押された時に動作するメソッドがあらかじめ定義されていますが，同様にいろいろな状況の変化(イベントという)に対応して起動されるメソッドが定義されています．イベントの例としては，マウスカーソルの通過やキーの押し下げなどがあります．

あらかじめParadoxに組み込まれ，定義されているメソッドを組み込みメソッドと呼びます．

このように，ボタンなどのUIObjectはイベントの生成で呼び出され，呼び出されると，イベントを解析し，イベントに対応する組み込みメソッドを呼び出します．

また，組み込みメソッドにはユーザーが指定する処理を組み込むこともできます．これを「アタッチする」と言います(**図3**)．

データの入力項目には「CanDepart」のメソッドがありますが，データ入力の終了などでカーソルが次の項目に移る前に呼び出されるメソッドです．

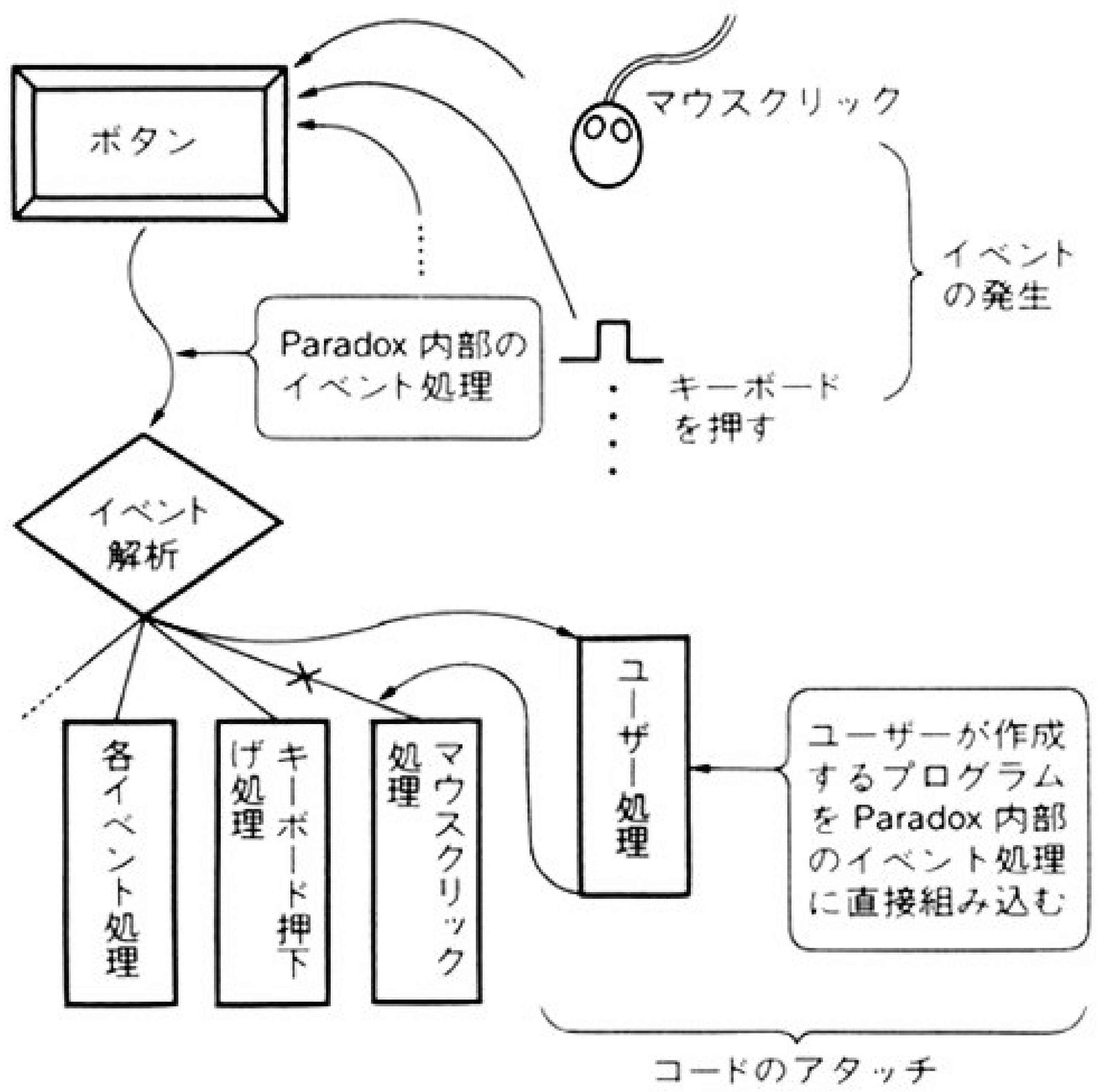
データの入力項目でエラーチェックを行うには，CanDepartにユーザーが作成したチェックルーチンをアタッチします．

このように，Paradoxでは処理を行う場合，従来のデータベース言語のようにメインの呼び出しから修正する必要がありません．

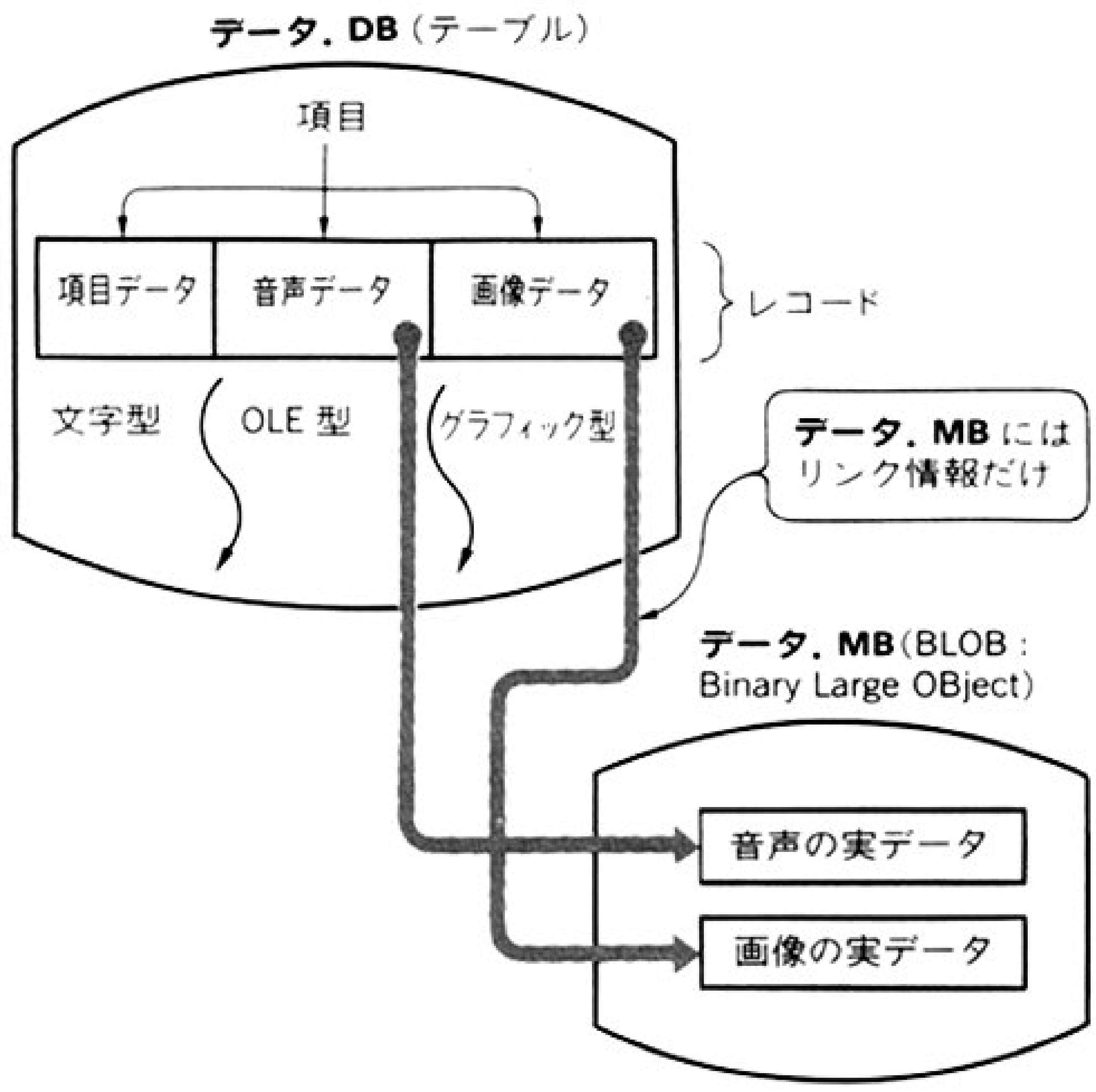
● **だから，このようにアプリケーションの開発ができる**

Paradoxは，設計に時間をかけ，最初から大規模なアプリケーションの開発もできます．しかし，アプリケーションを動作させながら，段階的に必要な機能を追加し，修正して完成させる手法に適していると思われます．

〈図3〉イベントと組み込みメソッド



〈図4〉データベースとファイルの構造



例えば，自分の管理する業務ならば，最初に対話形式で必要な項目を定義したテーブルを作成し，テーブルウィンドウからデータ入力/表示を行います．

またクイックフォーム（テーブルウィンドウで［F7］を押す）を使用すると簡単にフォームを作成できます．

フォームは1レコードのデータを1画面に表示でき，カードイメージでの画面表示/入力ができます．動作させながらクイックフォームを変更して，データ入力の項目に独自の入力処理を組み込んだり，キーマクロ風の処理をボタンに組み込むことによって，徐々に使いやすいものに作り上げていくことができます．

前述したように，ボタンを作成して処理を組み込んでも，いわゆるメイン処理のようなものは不要で，いきなりボタンと処理を組み込めます．

また，必要に応じて項目の追加もできるので，動作中でもシステムを作っていくことができます．

Paradox プログラムの説明

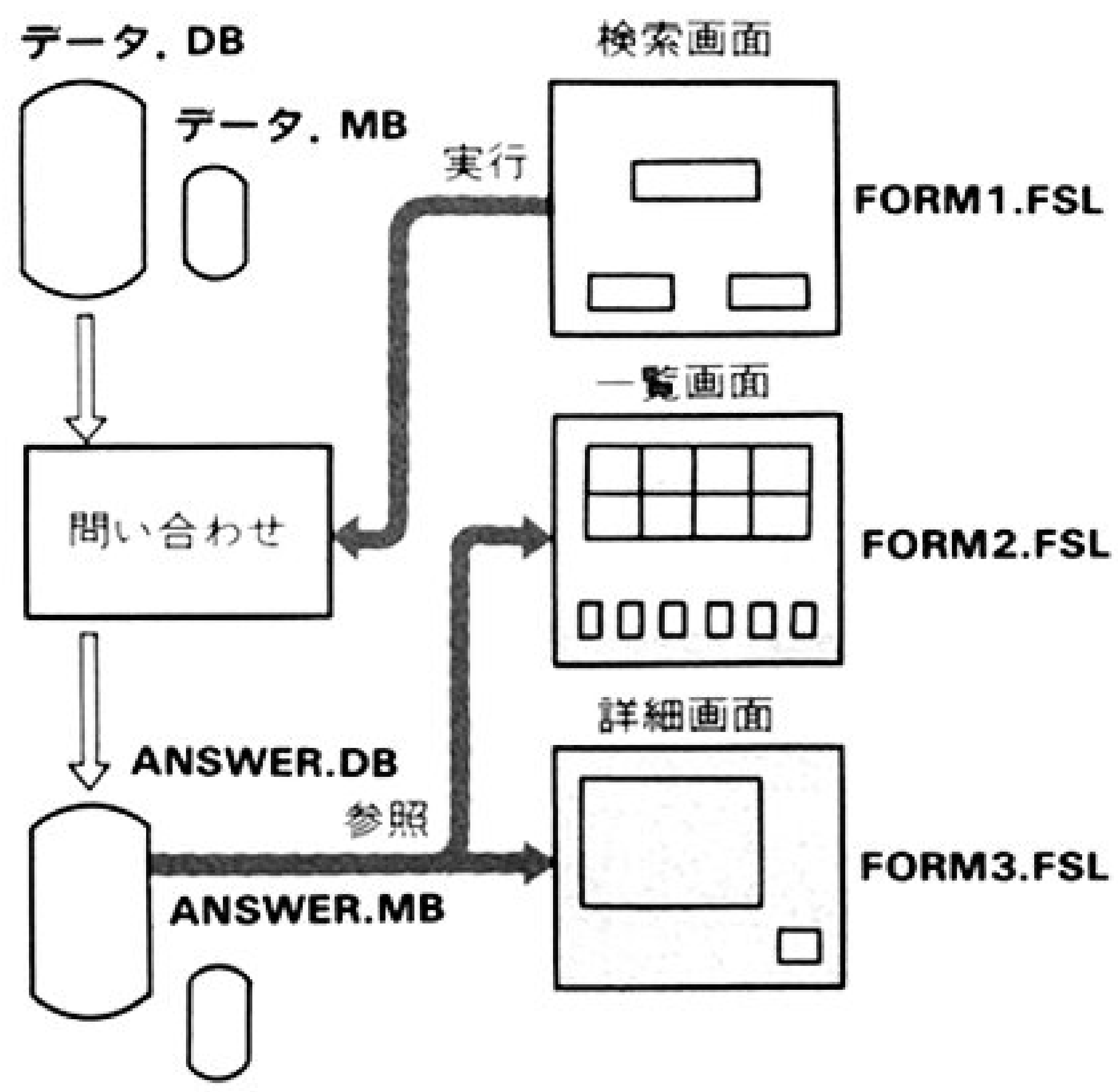
● データベースファイルの構造

図4が最初に紹介したプログラムのデータベースファイルの構造です．**データ.DB**に検索データなどの基本データを納め，**データ.MB**に画像/音声データを納めます．

Paradox はサイズが大きく，かつ不定のデータを別のファイルに格納します．これをBLOB（Binary Large OBject ファイル）と呼びます．これらのリンクの管理などは，Paradox が自動的に行います．

従来では，ユーザー自身が管理していた画像/音声といったこまごましたデータファイルをひとつにまとめてくれます．

〈図5〉プログラムとテーブルの関連図



ただし，データ量が非常に大きい場合は，ひとまとめのBLOBではなく個別のファイルで管理しないと，バックアップ運用やデータの差し替えなどがやりにくい場合もあります．

● 音声データ

Paradox は画像データを直接表示できますが，音声データを直接鳴らすことはできません．音声データはBLOB 上に格納され，読み出してから Windows の別プログラムであるサウンドレコーダに引き渡され音を出します．

サウンドレコーダを起動するメカニズムは Windows の OLE と呼ばれるしくみを使用します．Paradox のテーブル項目の種類に［OLE 型］があり，音声データを格納します．対話式操作でカーソルを音声データの上におき，項目表示のスピードバー（後述）をクリ

ックすると，サウンドレコーダが起動し音が鳴ります．

また，ObjectPAL から OLE を起動する機能があり，音声データをパラメータにしてサウンドレコーダを起動できます．

● プログラムの構成

図 5 がプログラムとテーブルの関連図です．検索画面で入力した検索データで**データ.DB** を問い合わせます．問い合わせ結果は，**データ.DB** から **ANSWER.DB** という一時ファイルにコピーされます．

● 起動時の画面

Paradox の操作について説明します．Paradox の起動画面は**図 6** のとおりです．

メニューバーと，スピードバーで表示される項目は，現在のウィンドウで使用できる種類だけが表示されます．本稿の内容を動作させて，メニューバーに指定の項目が見あたらないときは，カレントのウィンドウの種類が異なっていないかどうか確認してください．

マウスポインタをスピードバー上に置くと，その機能の説明をステータス行に表示します．

メニューバーから［プロパティ(P)］-［デスクトップ(D)］を選択するとデスクトッププロパテイを表示します．この中に「ObjectPAL レベル(L)」の設定があります(**図 7** 参照)．

これは，問い合わせのエディットウィンドウやデバッガウィンドウで，型やメソッド，プロシージャ，定数などのリストを表示する際にすべてを表示するか，初心者向きのものだけを限定表示するかを設定するものです．上級者用を選択しておけばよいでしょう．

● テーブルの表示

テーブルの内容を表示する場合，通常はテーブルウィンドウを使用します．テーブルウィンドウ以外では，フォームまたはレポートを使用する方法もあります．

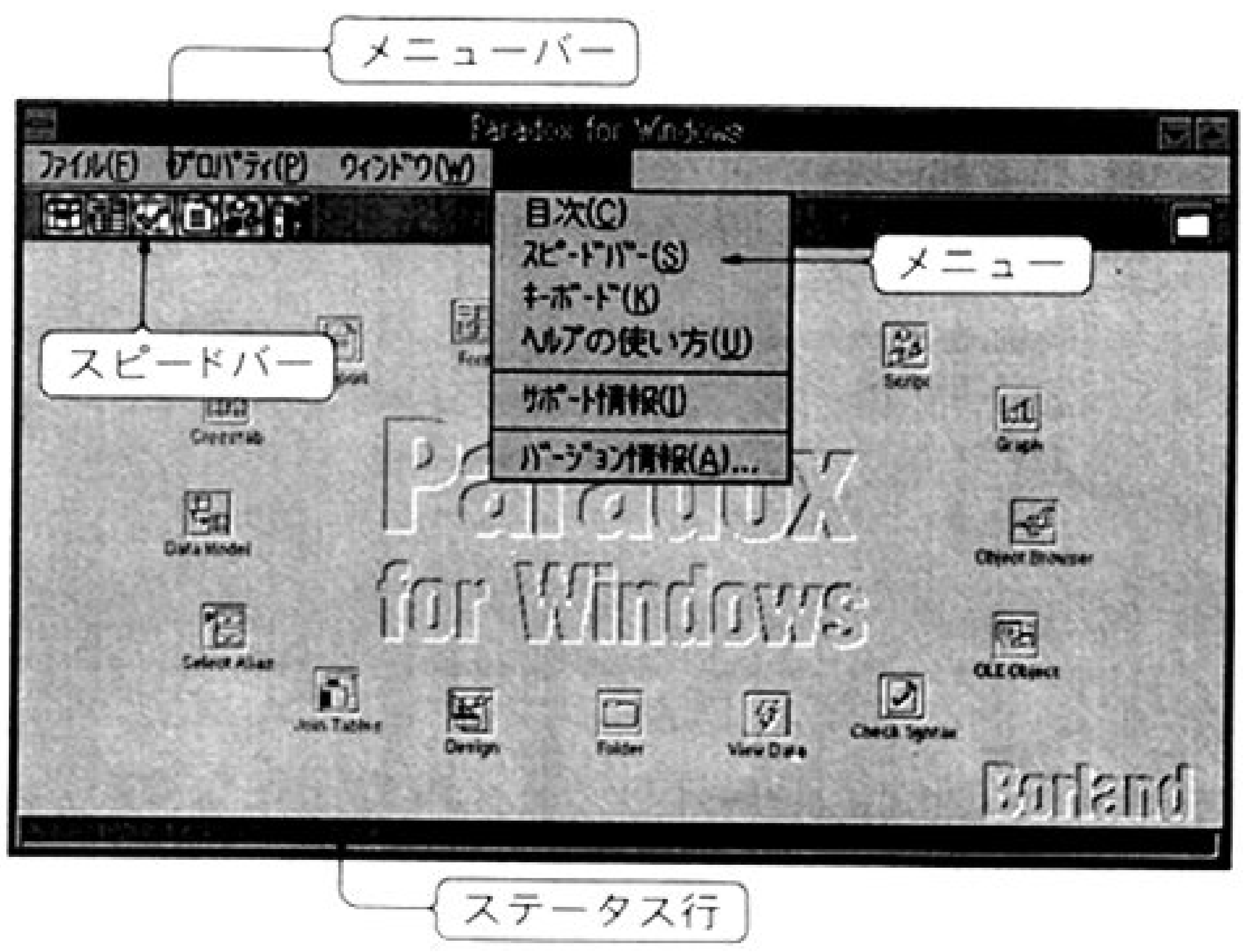
フォームは，あらかじめ作成しておくので，プログラムに組み込む場合とか，何度も表示をさせる場合に使用します．前述のプログラムではフォームを使用しています．レポートは印刷を目的とするものです．

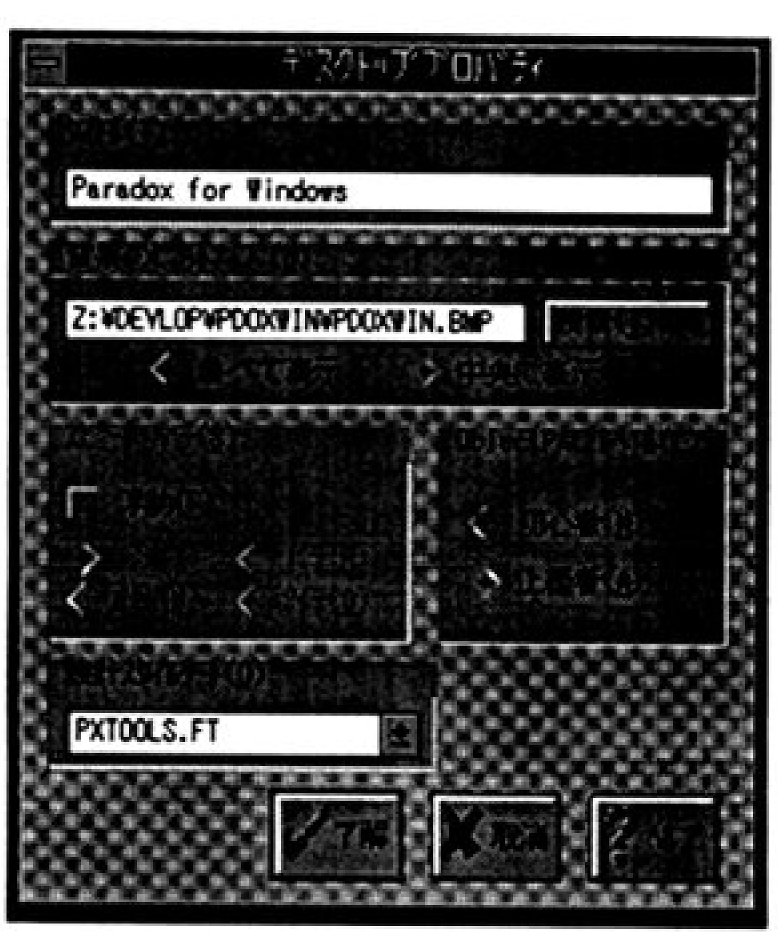
テーブルウィンドウは，テーブルの 1 レコードを 1 行に表示します．表示項目の表示幅，高さは，マウスでドラッグすることにより変更でき，変更内容をセーブできます．タイトル，項目の表示属性はテーブルのプロパティ(後述)で変更できます．

テーブルウィンドウから，［F7］を押すとクイックフォームを表示し 1 画面/1 レコードのカード形式で表示します．クイックフォームは，Paradox が作成する規定値のフォームでユーザーが自由に変更できます．

項目の配置やサイズなどを変更し，名前を付けてフォームとしてセーブできます．またボタンなどの配置

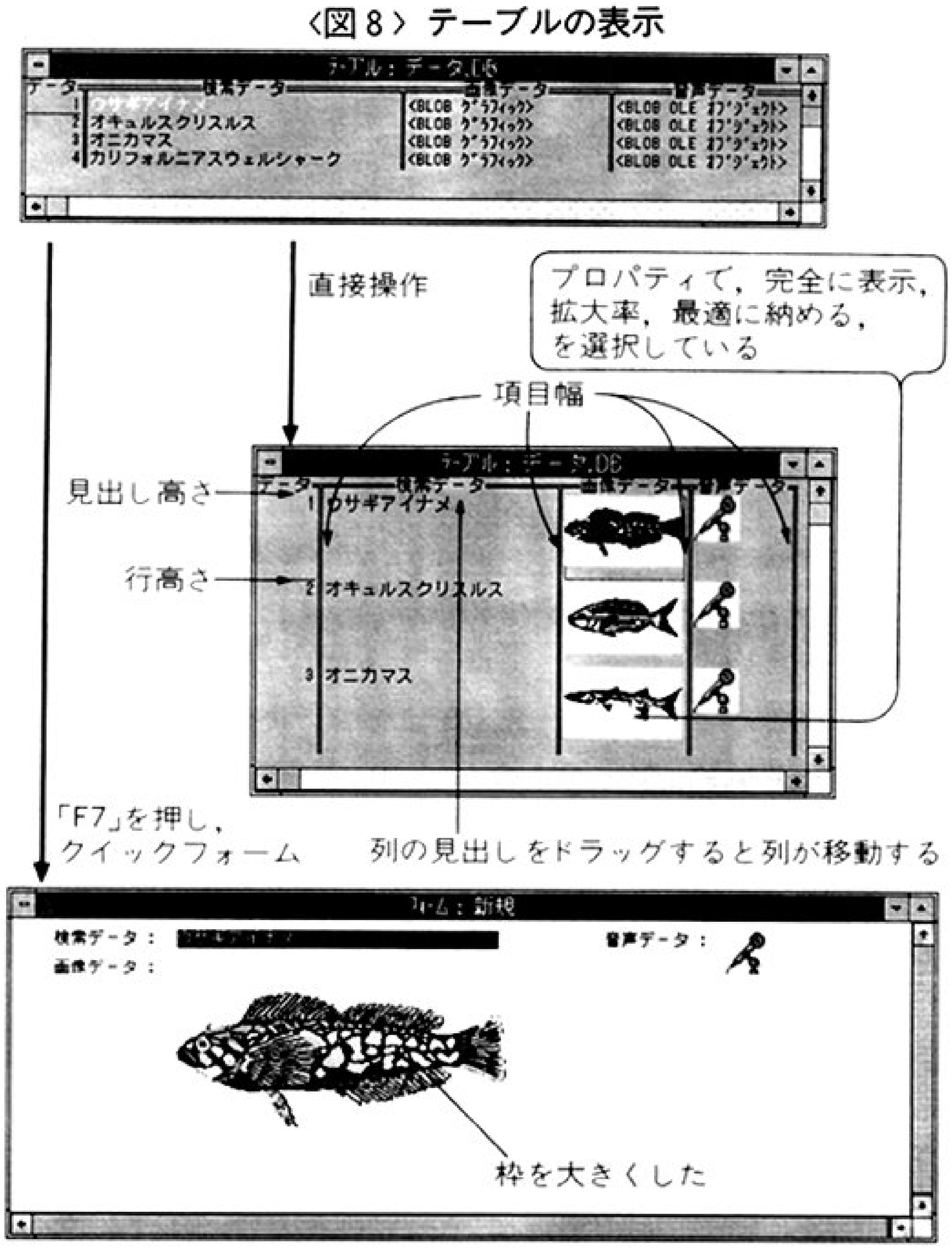
〈図 6〉Paradox の起動画面





〈図 7〉
ObjectPAL の
レベルの設定画面

〈図 8〉テーブルの表示



もできます(**図 8**).

なお，テーブルウィンドウはフォームではないのでボタンを配置できません．ボタンを配置する場合は新しいフォームを作成し，テーブルフレームを配置するとテーブルウィンドウ風の一覧表を表示できます．

● テーブルウィンドウの操作

テーブルウィンドウを開くには，メニューバーから ファイル(F) - 開く(O) - テーブル(T) を選択するとテーブル名を問い合わせててきます．テーブル名を入力するとテーブルウィンドウが開いて，一覧形式でテーブルを表示します．

テーブルウィンドウから，→，←，↑，↓ および ROLLUP，ROLLDOWN で，DOS/V 機の場合なら PageDown，PageUp を使用して項目を移動できます．

項目の表示枠が，データの長さより短い場合，項目表示(後述)で左右スクロールを行うと内容が確認できます．

項目値を変更する場合，F9 または，スピードバーの データ編集 を押すと編集モードになります．

編集モードでは，Ctrl + D ですぐ上のレコードのデータコピー，Ins で空白レコードの挿入，Ctrl + Del でレコード削除，ESC で現レコードの編集中のデータを元に戻します．

項目データを編集する場合，BS を押すと元の内容が消去され，再入力できます．元の内容を直接編集する場合，F2 またはスピードバーから項目表示を押し，項目表示に入ります．項目表示では →，← でカーソル位置が項目内で左右に移動し，項目を編集できます．

● テーブルウィンドウのプロパティ

テーブルウィンドウの各プロパティを変更すると，テーブルウィンドウに表示しているタイトルとデータを分ける罫線，見出し，各項目の表示方法，現在のレコード表示の色，種類を個別に変更できます．

例えば，金額で 1000 円以上は赤色表示するとか，10 歳以下の子供は黄色で表示するということなどができます．画像項目では縮小率，すべての表示を指定できます．

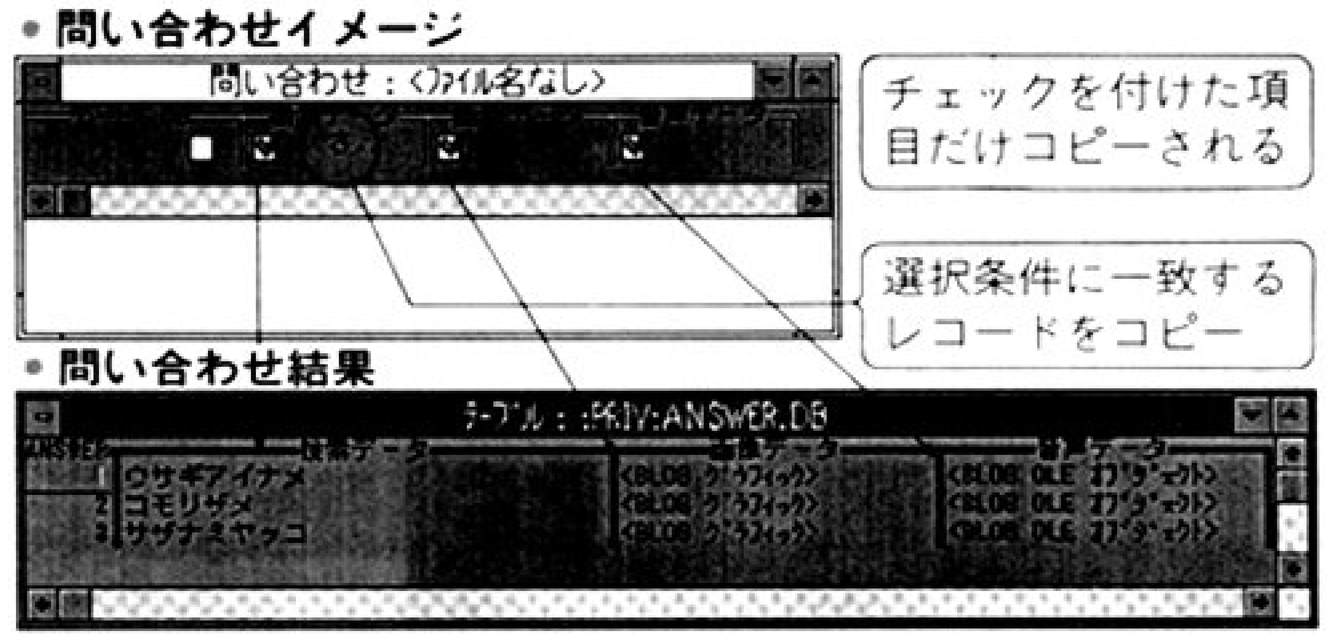
プロパティの変更は，変更したい項目にカーソルを位置付けて右クリックをするか，またはメニューバーから プロパティ(P) - データ(A) を選びプロパティメニューから変更します．

変更した(上書き保存を選んだ)プロパティの情報は，テーブル名が同一で拡張子が **TV** のファイルに保存されます．変更したプロパティを初期値に戻すには，メニューバーから プロパティ(P) - 表示プロパティ(V) - 削除(D) を選びます．

● 問い合わせ

問い合わせは，テーブルから必要なデータを検索し

〈図 9〉問い合わせの選択条件を設定



て，条件の一致したデータを抜き出します．

Paradox は，QBE(Query By Example)という方法で問い合わせを行います．いろいろな機能がありますが，ここでは前述のプログラムで必要な機能を説明します．

図 9 の問い合わせのイメージから問い合わせの選択条件を設定し，問い合わせを実行すると，選択条件に一致したレコードでチェックマークが付いている項目を **ANSWER.DB** にコピーします．

問い合わせのイメージは，問い合わせを行うテーブルと同じ構造です．選択条件はおのおのの項目に設定できます．

メニューバーから ファイル(F) - 新規作成(N) - 問い合わせ(O) を選択すると，対象テーブル名を問い合わせてきます．テーブル名を入力すると問い合わせウィンドウが開いて，問い合わせのイメージを表示します．

問い合わせウィンドウで検索結果に必要な項目にチェックをつけます．チェックはトグル式になっており，再びチェックするとクリアされます．

問い合わせ条件はファイルに保存でき，そこから呼び出すこともできます．また，保存した問い合わせはテキストファイルなので，プログラムで問い合わせ文を作成する場合のひな形としても使えます．

保存するには，問い合わせウィンドウのメニューバーから ファイル(F) - 名前をつけて保存(A) を選択しファイル名を入力して実行します．

問い合わせの選択条件で，**@**は任意の 1 文字をあらわし，**..**は任意の長さの文字を表します．例えば名前の項目などで**タ@@**は**タナカ**，**タヨシ**などと一致します．また**タ..**は**タナカ**，**タマガワ**などと一致し**..タ..**は**タ**が含まれるすべて文字と一致します．

● 作業ディレクトリと専用ディレクトリとエイリアス

Paradox が使用するテーブルや，オブジェクトが入っているカレントのディレクトリを作業ディレクトリといいます．

また，一時テーブルを作成するデレクトリを専用ディレクトリといい，問い合わせ結果の **ANSWER.DB** などが作成されます．

プログラムマネージャのアプリケーションアイコンに登録されている実行時のディレクトリは，自動的に作業ディレクトリにはなりません．通常，作業ディレクトリはParadoxが記憶していて，前回終了時の状態を引き継ぎます．

専用ディレクトリはネットワークなどのマルチユーザー環境で使用する場合，他のユーザーと干渉しないようにユーザー自身のディレクトリを設定します．

ネットワークを接続していない場合は，適当なディレクトリを指定します．専用ディレクトリに作成する一時テーブルは，Paradox終了時に削除されます．

一時テーブルを保存する場合はParadoxの終了前に名前を変更します．テーブル名の変更は関連ファイルの変更もともなうので，DOS上やWindows上で行わずに，必ずParadox上で行います．メニューバーから［ファイル(F)］-［ユーティリティ(U)］-［名前の変更(R)］を選択して行います．

ディレクトリ名にはエイリアスが使えます．作業ディレクトリは **:WORK:**，専用ディレクトリには **:PRIV:** というエイリアスが割り付けられています．例えば専用ディレクトリに **C:¥AAA¥BBB** が割り付けられていて，専用ディレクトリの **ANSWER.DB** を参照する場合，**:PRIV:ANSWER.DB** と指定すると **C:¥AAA¥BBB¥ANSWER.DB** をアクセスします．

問い合わせをした場合，結果の **ANSWER.DB** は専用ディレクトリに作成されるので，参照をする場合は，**:PRIV:ANSWER.DB** と指定します．

● テーブルの作成

メニューバーから［ファイル(F)］-［新規作成(N)］-［テーブル(T)］を選択したあと，［新規作成(N)］を選択します．すると，テーブルタイプを問い合わせてくるので，規定値の **Paradox for Windows** を選んで［了解］をクリックします．

また，既に作成したテーブルの構造を変更する場合は，［ユーティリティ(U)］-［再構築(T)］を選択してファイル名を入力します．

図10 のように，テーブルの作成ウィンドウで項目名，型，幅を入力します．型はスペース(またはマウスの右ボタンでクリック)で一覧を表示し，そこから選びます．文字型のデータでは幅は必要ですが，それ以外の型では幅は特に必要ではありません．キーも不要です．

必要なデータを入力後に［新規保存(A)］を選択すると，ファイル名を問い合わせてきます．今回は **データ.DB** とするので，**データ** を入力します．

● フォームの作成

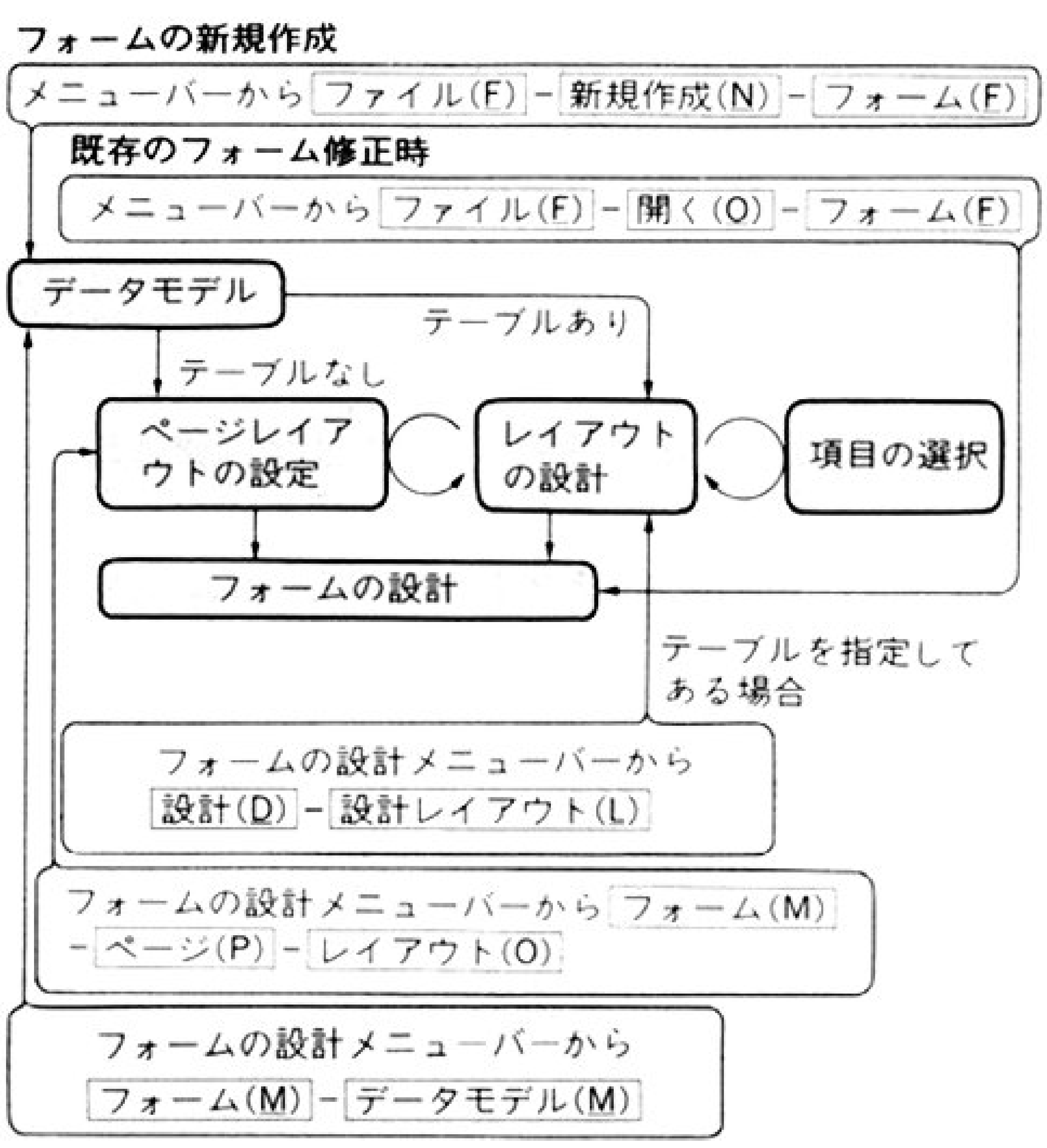
図1 からもわかるように，**FORM1**，**FORM2**，**FORM3** の三つのフォームを作成します．

FORM1 はテーブルの項目を使用しません．**FORM2** と **FORM3** は **ANSWER.DB** の項目を使用します．**ANSWER.DB** は **FORM1** の検索のボタンメソッドが作成します．対話操作で作成するか(**図9** 参照)，**FORM1** を動作させるかして **ANSWER.DB** を作成してください．

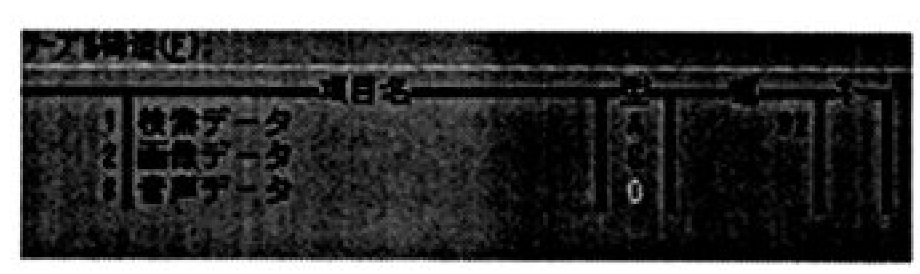
フォーム作成は，設定する事項が多く，数種類のウィンドウから入力して作成します．**図11** の各設定ウィンドウの関連図を参照してください．

フォームを作成するには，メニューバーから［ファイル(F)］-［新規作成(N)］-［フォーム(F)］を選択します．すると **図12** のデータモデルのウィンドウが開いて，フォームと結合するテーブルを問い合わせてき

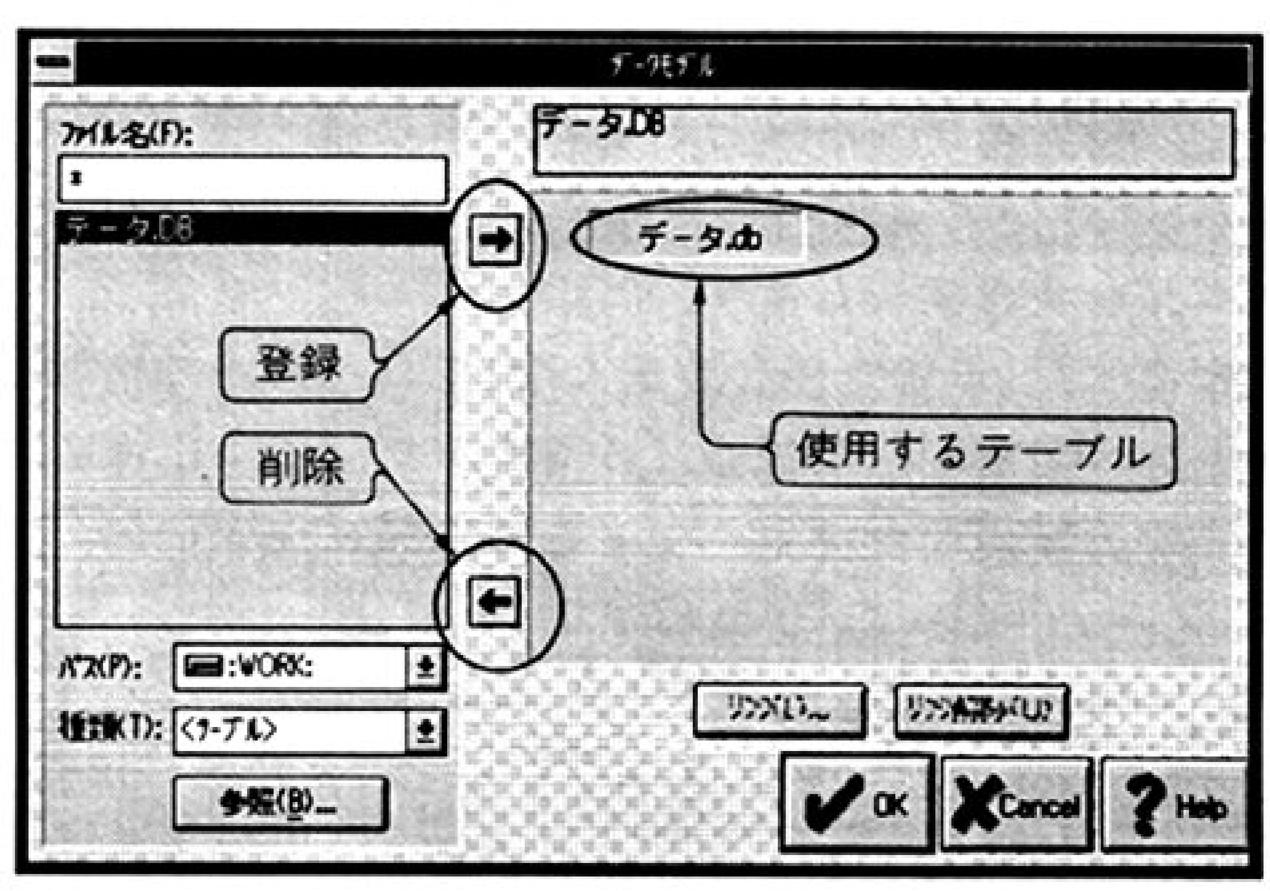
〈図11〉各設定ウィンドウの関連図



〈図10〉テーブル作成ウィンドウ



〈図12〉データモデル



ます．

FORM1はテーブルを使用しないので，そのまま**FORM2**と**FORM3**は**データ.DB**を選んで［了解］を選びます．

テーブルを指定した場合は，**図13**の「レイアウトの設計」のウィンドウが開きます．テーブルを指定しない場合は，**図14**の「ページレイアウトの設計」のウィンドウが開きます．

図13の「レイアウトの設計」から，［項目の選択(E)］を選ぶと**図15**の［項目の選択］画面を表示し，フォームに配置するテーブルの項目を指定できます．

FORM2の場合は，「レイアウトの設計」の「スタイル(S)」から「マルチレコード(M)」を選んでください．

作成したフォームを修正する場合，メニューバーから［ファイル(F)］-［開く(O)］-［フォーム(F)］を選択し(**図16**)，ファイル名を入力するか，または一覧から選択します．

● フォームのレイアウト配置

ボタン，文字などはオブジェクトと呼びますが，オブジェクトは，スピードバーから選んで，フォームに配置します．例えば，ボタンを配置する場合，スピードバーからボタンツールを選び，フォーム上でドラッグします．

スピードバーには，文字を配置するテキストツール，ボタンを配置するボタンツール，項目を配置する項目ツールなどがあります．項目ツールは，テーブルと結合する場合と，テーブルと結合しない単なる入力/出力用の領域の場合とがあります．

オブジェクトはマウスで位置，サイズを変更できます．余分なオブジェクトを削除する場合，選択して［DEL］を押します．

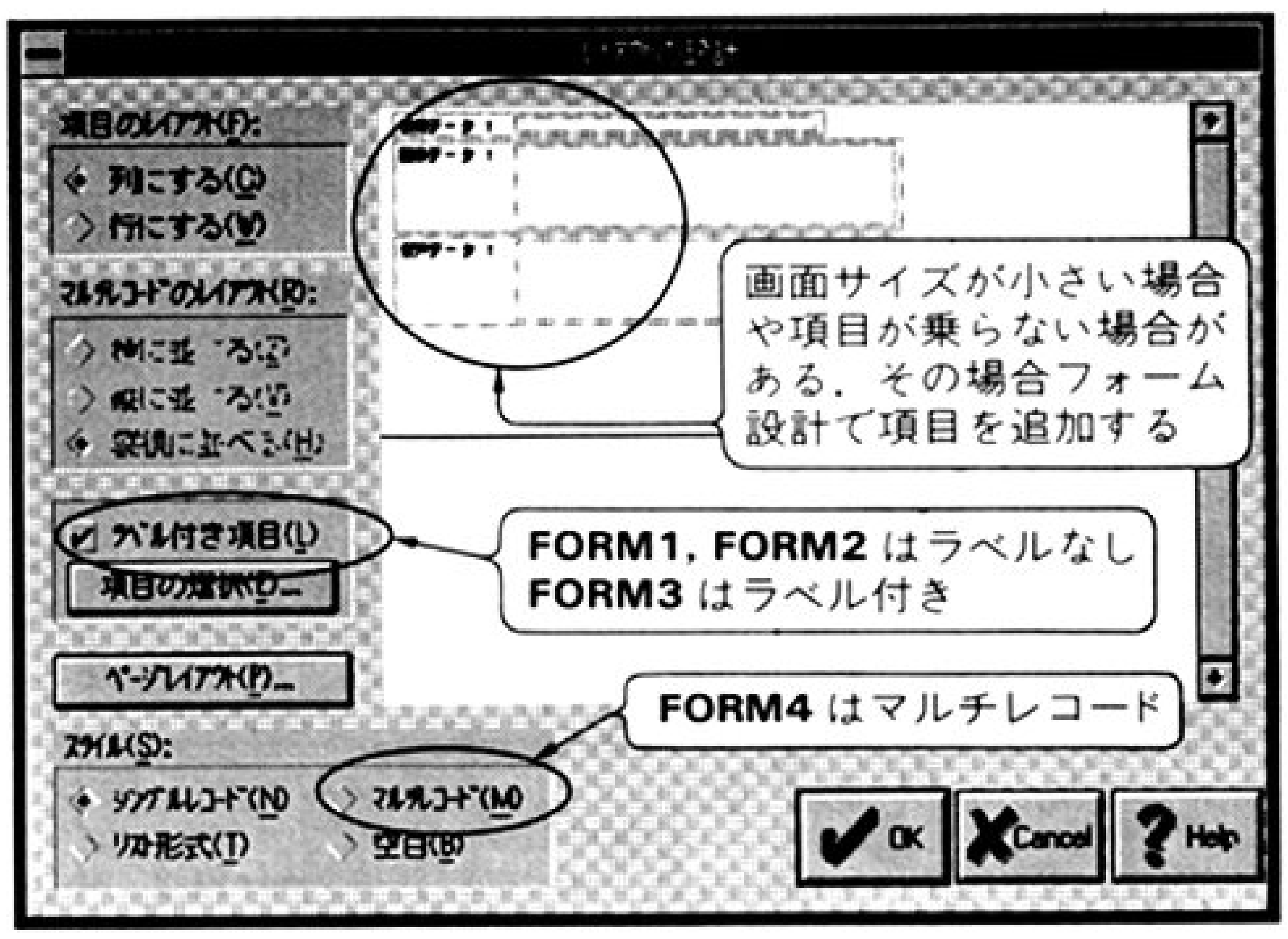
配置したオブジェクトは，プロパティを設定します．オブジェクトの上で右クリックをするとプロパティのメニューを表示します(**図17～図20**)．

FORM2はマルチレコードです．先頭のレコードのオブジェクトで項目などを定義します．残りのレコードオブジェクトは先頭と同じ内容で定義されます．

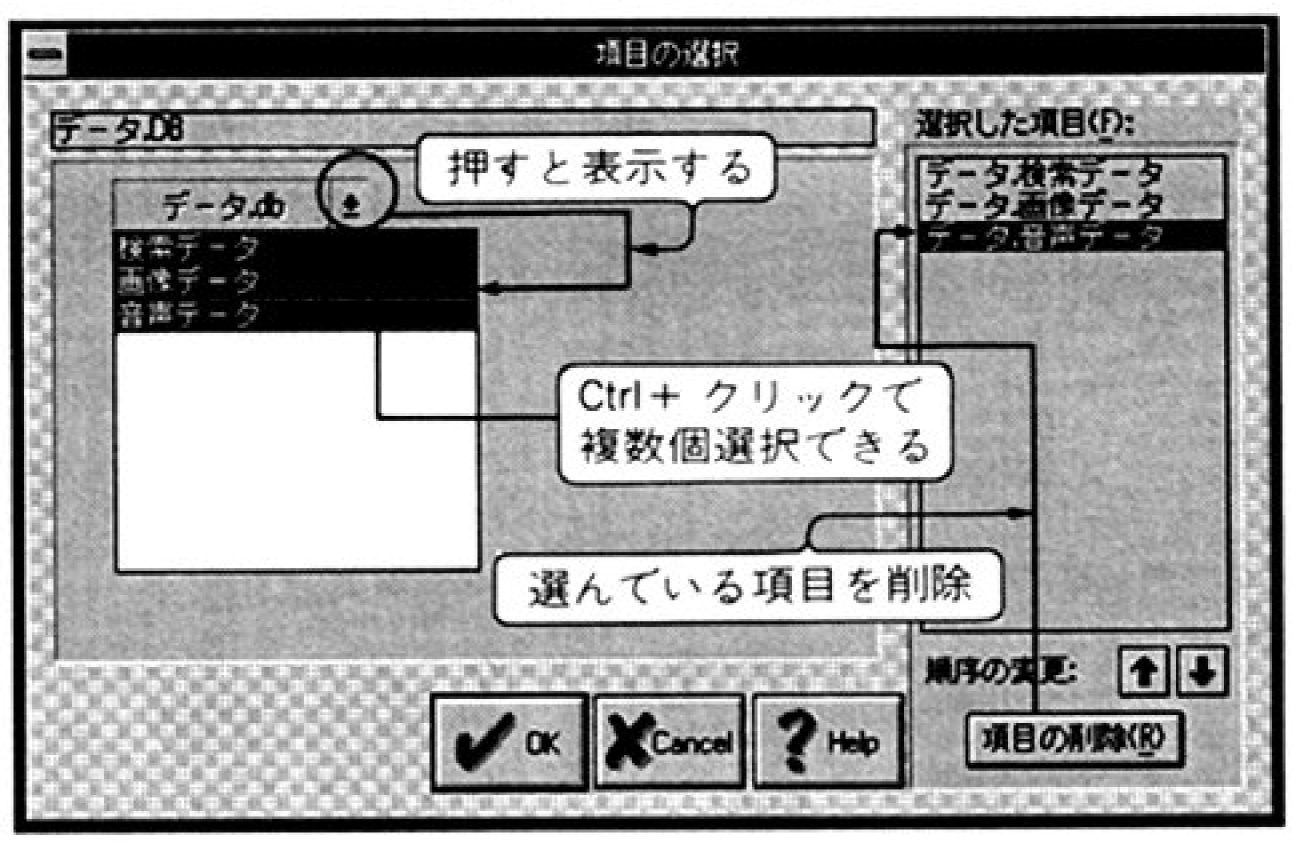
各オブジェクトのサイズを調整し，レコードオブジェクトのサイズを4列2段に納まるサイズにして，マルチレコードで4列2段を指定します．

フォーム，ボタンなどのオブジェクトには，Paradoxが名前(通し番号)を付けますが，メソッドから参照される場合，名称は意味のある名前に変更したほうがプログラムが読みやすくなります．今回**FORM1**の入力項目は名前を“項目検索データ”に変更しました．同じように**FORM3**を作成してください．

〈図13〉 レイアウトの設計



〈図15〉 項目の選択





〈図14〉 ページレイアウトの設計



〈図16〉 設計文書を開く

● **データの登録**

Paradox を対話形式で使用してデータを登録します．使用パソコンに録音機能がある場合はマイクから録音した音を登録できます．音声の登録は Windows のサウンドレコーダを使用します．サウンドレコーダと Paradox を切り替えながらクリップボード経由でコピーします．

具体的には，Paradox のメニューバーから［ファイル(F)］-［開く(O)］-［テーブル(T)］で**データ.DB** を表示，［F9］を押して，編集モードに入ります．データを追加する場合は［INS］を押して新しいレコードを作成します．

まず検索データを入力し，次に音声データの項目に入ります．そこで［GRPH］＋［ESC］，DOS/V 機の場合なら［Alt］＋［ESC］を押すと，アプリケーションが切り替わります．

サウンドレコーダで，音声データ(**.WAV** のファイル)を取り込むとかマイクで録音するなどして，音声をサウンドレコーダ内に読み込みます．

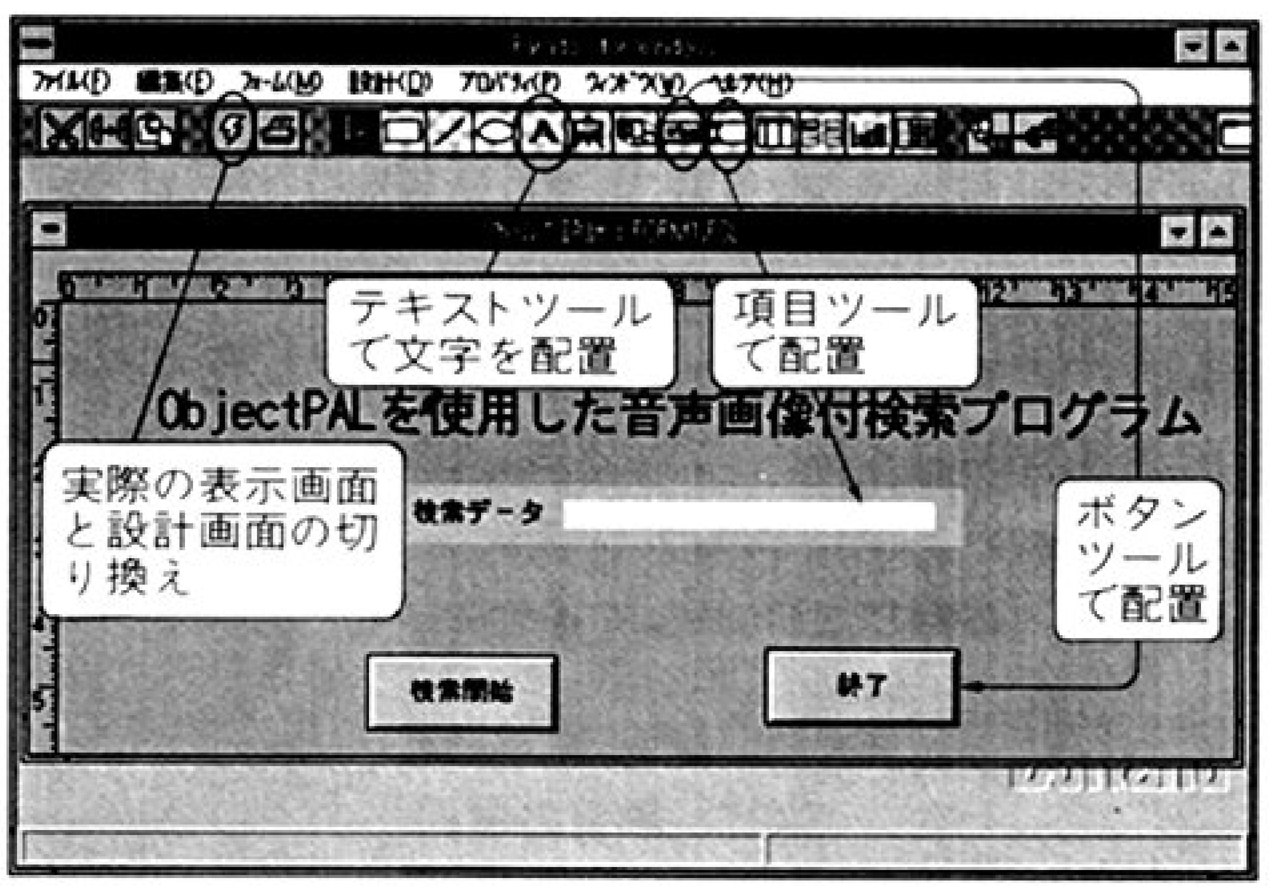
次にサウンドレコーダのメニューバーから［編集(E)］-［コピー(C)］を選び，データをクリップボードにセットします．

このクリップボードにデータがセットされている状態で［GRPH］＋［ESC］を順次押し，アプリケーションを Paradox に切り替えます．

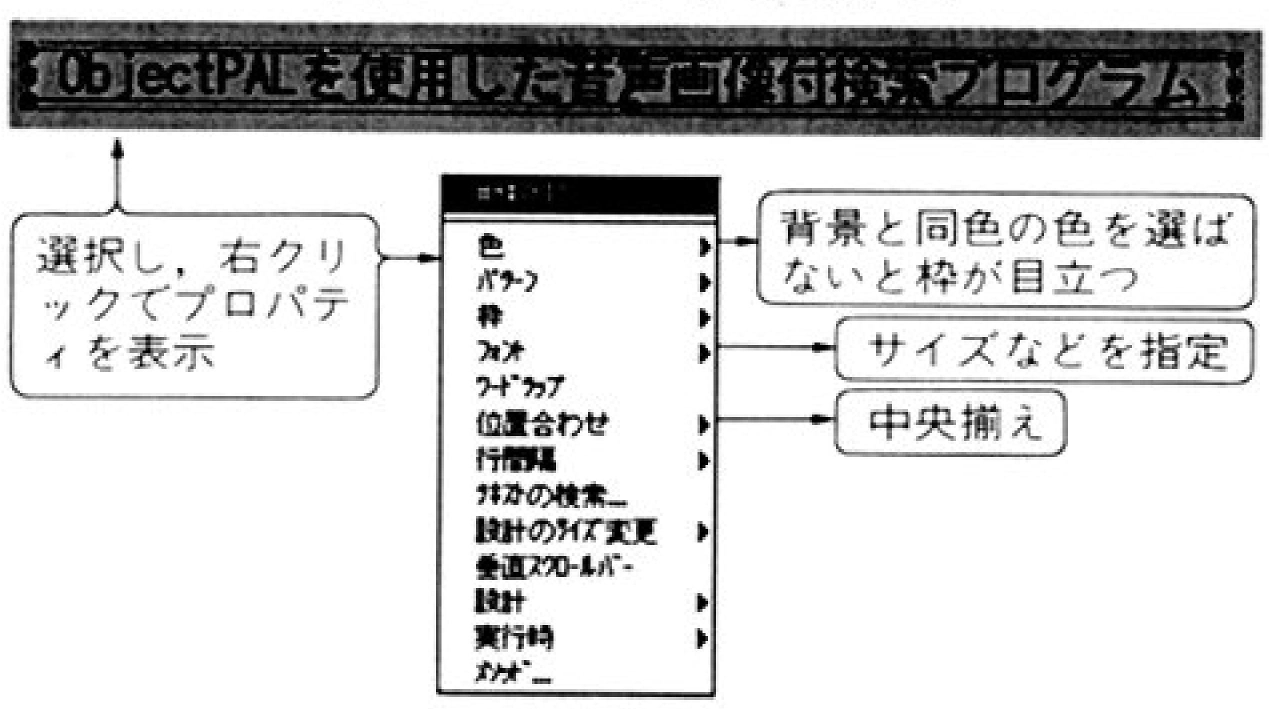
Paradox のカレントのウィンドウがデータのテーブルウィンドウで，項目が音声データで編集中であるのを確認し，メニューバーから［編集(E)］-［貼り付け(P)］を選び，データをクリップボードから貼り付けます．

画像データは画像データファイルかクリップボード経由で貼り込めます．画像データファイルの場合，メニューバーから［編集(E)］-［指定元から貼り付け(F)］を選び，画像データファイルから貼り付けます．貼り付けると画像データはテーブルにコピーされるので，

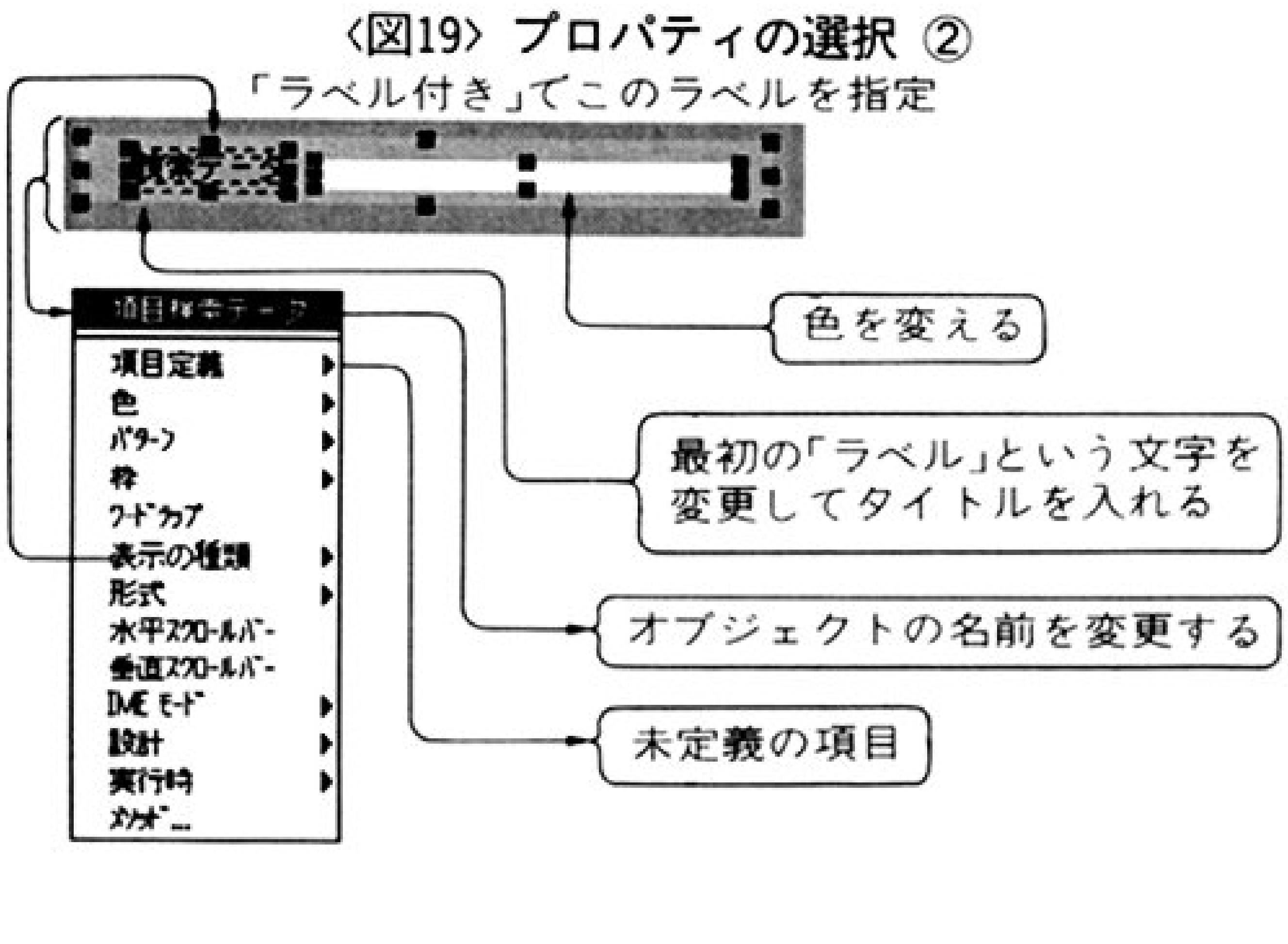
〈図17〉 フォームの設計



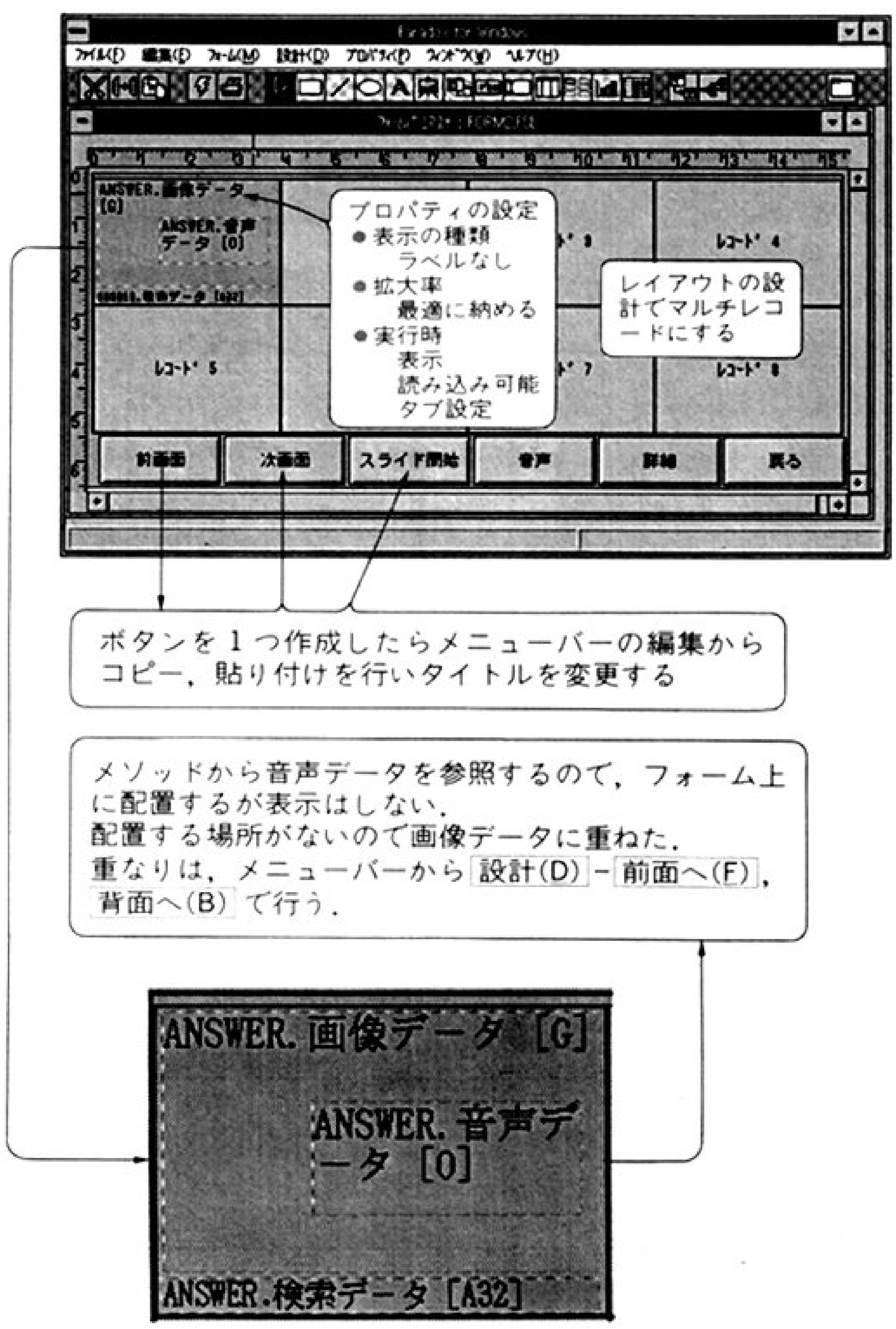
〈図18〉 プロパティの選択 ①



〈図19〉 プロパティの選択 ②



〈図20〉 レコード



元のファイルは不要です.

クリップボード経由で貼り付ける場合は，ペイントブラシなどの画像処理プログラムでクリップボードに画像データをセットして，メニューバーから[編集(E)]-[貼り付け(P)]を選んで貼り付けます.

● メソッドの入力

メソッドを入力する時は，ObjectPALエディタを使用します.オブジェクトのプロパティメニューから「メソッド...」を選ぶとメソッドダイアログが開きます.そして，「組み込みメソッド」から目的のメソッドを選ぶとObjectPALエディタが開きます.

このエディタは基本的な機能しかないので，長いソースを入力する場合は別に作成したテキストを読み込むか，使い慣れた別のエディタを組み込んだほうがよいでしょう.

別のエディタを組み込む場合は，ObjectPALエディタのメニューバーから[プロパティ(P)]-[別のエディタを選択(A)]から外部のエディタプログラム名を登録します.

別のエディタを起動した場合，内蔵エディタに比べると制約があり，エディタを終了しないとParadoxには戻れません.そのため複数のメソッドを同時に開いての編集はできません.当然，構文チェックはParadoxに戻ってから行います.

構文チェックを行うには，ObjectPALエディタのメニューバーから[言語(L)]-[構文チェック(C)]を選びます.

● ObjectPAL デバッガ

エディタウィンドウからソース入力後，構文チェックを行います.構文チェックがOKならば，ObjectPALデバッガを使用してデバッグします.

デバッグは，ブレークポイント，またはソースプログラムのデバッグ文〔**DEBUG( )**〕を設定して，実行中のプログラムをデバッガに戻し，変数の内容の表示/変更，トレースなどを行います.

ObjectPALエディタのメニューバーから[デバッグ(D)]を選択するとメニューを表示します.メニューから[ブレークポイントの設定(S)]，または[DEBUG文を有効にする(E)]などを設定して実行させます.

ボタンなどを押して，デバッグをするメソッドのイベントを発生させて，実行を中断させます.ブレークポイントなどで停止するとエディタウィンドウが開き，停止したソース位置を表示します.後は，変数を調べたり，プログラムをシングルステップで動作させたりしてデバッグを続けます.

Paradox 簡単な文法説明

ObjectPALの構文は自由形式で，スペースで区切れば132バイトの1行に複数文を記述できます.シンボルは半角文字の大文字/小文字の区別はしません.

コメントは;以降行末までか，中カッコ { } で囲まれた部分です.

制御文には **if**, **switch**, **iif**, **forEach**, **loop**, **scan**, **while**, **quitLoop**, **return** などがあります.このうちの **iif** はC言語の **a?x : y** の条件演算に相当し，**forEach** は動的配列の各要素に対する処理を繰り返します.

データ型は，最初から固有のメソッドを持っています.例えば，**リスト1**では，文字列の変数を定義し，その文字列型のメソッド **view** を使用して，変数の内容を表示/入力するダイアログボックスを表示します.す.

〈リスト1〉データ型は固有のメソッドをもつ

```
method run(var eventInfo Event)
  var
    st     string ;文字列型変数 st 定義
  endvar
  st="テストデータ"
  st.view()
endmethod
```

使用できる演算子を次に示します.

四則演算　：**+**，**-**，*****，**/**
比較演算子：**=**，**<>**，**<**，**<=**，**>**，**>=**
論理演算子：**AND**，**OR**，**NOT**

ビット演算は次のメソッドで行います.

bitAND( )，**bitOR( )**，**bitXOR( )**

Paradox より本格的なプログラムにするには

今回の作成プログラムはどちらかというと学習用です.ObjectPALを習得した方なら，本格的なプログラムに改造してみてください.

使用者が必要なテーブル項目を追加するのはもちろん，一覧表示用の縮小画像の項目を別に持たせるほうがよいでしょう.

フルサイズの画像を毎回アクセスすると処理に時間がかかります.問い合わせ結果のファイルにBLOBの項目を入れると，問い合わせ時にデータ量の多いBLOBの項目をコピーするため，同じように時間がかかります.一覧表示側で，元のテーブルから表示すべきでしょう.

また，テーブルファイルは，部分一致を行うため，キーを付けていません.やはり，検索データの項目数を複数にして，第1キーなどで範囲を絞り，その中で，必要ならば部分一致で探すようにすべきでしょう.

*　　*　　*

筆者は仕事を探しています.
よろずソフト開発承ります.
連絡先☎ 078-753-6443　船江まで.

高速シリアルインターフェース用ドライバの制作

② 通信用インターフェースLSI 16550Aについて

Fast Serial

石川 治

今回は，超高速シリアルインタフェース・ボード MC16550 II〔㈱マイクロコア〕と，高速 RS-232C 拡張ボード RSB-384〔メディアインテリジェント㈱〕に使用されている 16550A(UART)ついて解説します．

16550A(UART)のレジスタ

はじめに，16550A のレジスタについてまとめておきます．

● **受信バッファレジスタ RBR**(Read only)

ライン・コントロールレジスタ LCR が **DLAB=0** のとき，受信データを入力します(受信レジスタ)．

● **送信ホールディングレジスタ THR**(Write only)

LCR が **DLAB=0** のとき，送信データを出力します(送信レジスタ)．

● **割り込み許可レジスタ IER**(Read/Write)

LCR が **DLAB=0** のとき，各種の割り込み許可条件を設定します［**図2-1**(a)］．

● **割り込み ID レジスタ IIR**(Read only)

割り込みが発生したとき，その割り込み要因を参照します［**図2-1**(b)］．

▶受信ラインステータス割り込み：オーバーラン，パリティ，フレーミングエラー，およびブレーク受信で発生．LSR を読むことでリセット．

▶受信データ有効割り込み：FIFO モードでは，受信されたデータが設定されたトリガレベルに達したとき発生．データを読み込むことでリセット．

▶受信データ・タイムアウト割り込み：FIFO 内に受信データがあるのに一定時間読み込まれなかった場合に発生．これもデータを読み込むことでリセット．

▶送信レジスタ空割り込み：FIFO モードでは，送信 FIFO が空になったとき発生．IIR を読むか，データを 1 バイトでも書き込むことでリセット．FIFO モードでは 16 バイトまで一気に書き込み可能．

▶モデムステータス割り込み：MSR を前回読んだ以降，CTS, DSR, RI, DCD に変化があった場合に発生．MSR を読むことでリセット．

割り込みの優先順位は，図の( )内の数値の小さいほうが高くなります．

なお，割り込み要因が発生していても，IER の対応する許可ビットが設定されていなければ，IIR の要因ビットもマスクされて読み取れませんので注意してください．

● **ライン・コントロールレジスタ LCR**(Read/Write)

回線制御用レジスタで，通信パラメータなどを設定します［**図2-1**(c)］．

● **モデム・コントロールレジスタ MCR**(Read/Write)

RS-232C 信号線により，モデムをコントロールします．

MCR については，MC16550 II と RSB-384 とでは仕様が若干異なります［**図2-1**(d)］．MC16550 II ではループバックテスト・ビット **LOOP** が有効ですが，RSB-384 では必ず 0 に設定することになっています．

一方，汎用出力ビットは MC16550 II では無効ですが，逆に RSB-384 では $\mathbf{Out_1}$，$\mathbf{Out_2}$がともに有効です．

ここで特に注意しなければならないのは，RSB-384 で割り込み制御を行う場合，$\mathbf{Out_2}$(IEN)が割り込みラインの制御を直接行っているということです．このビットを 1 にしておかなければ，割り込みがかかりません．

これは IBM PC の非同期通信アダプタの仕様にならったものと思われますが，8251A から移行してドライバを制作する場合，多少とまどう点だと思います．

● **ライン・ステータスレジスタ LSR**(Read/Write)

エラー発生などの通信状況を取得します［**図2-1**(e)］．

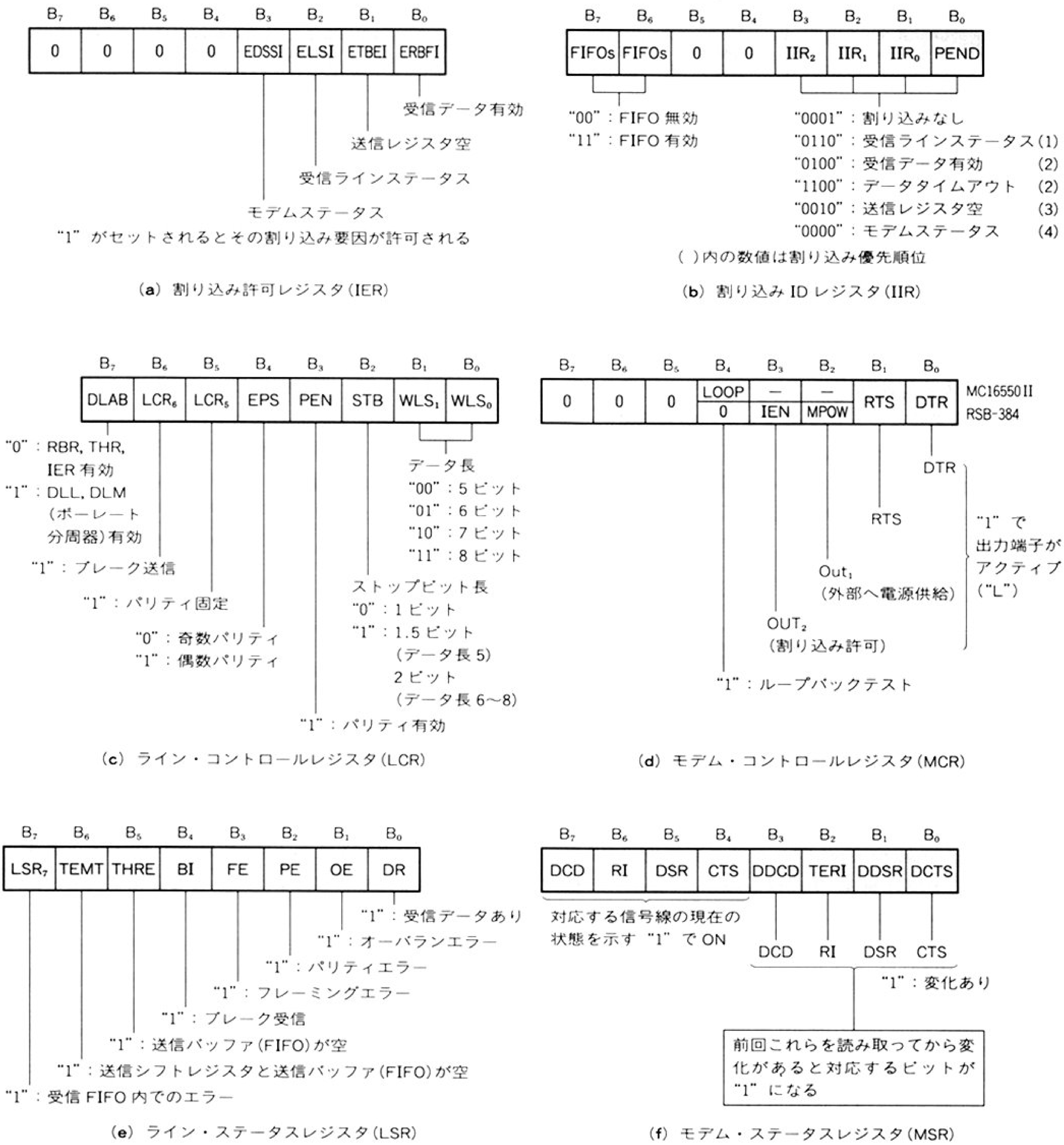
OE, PE, FE, BI のどれかひとつでも 1 になると，受信ラインステータス割り込みが発生します．$\mathbf{LSR_7}$も含めてこれらのビットは，本レジスタを読み込むことでリセットされます．

● **モデム・ステータスレジスタ MSR**(Read/Write)

RS-232C 信号線の状態を取得します［**図2-1**(f)］．

DCD, RI, DSR, CTS の各信号線の状態は，正論理で示されることに注意してください．

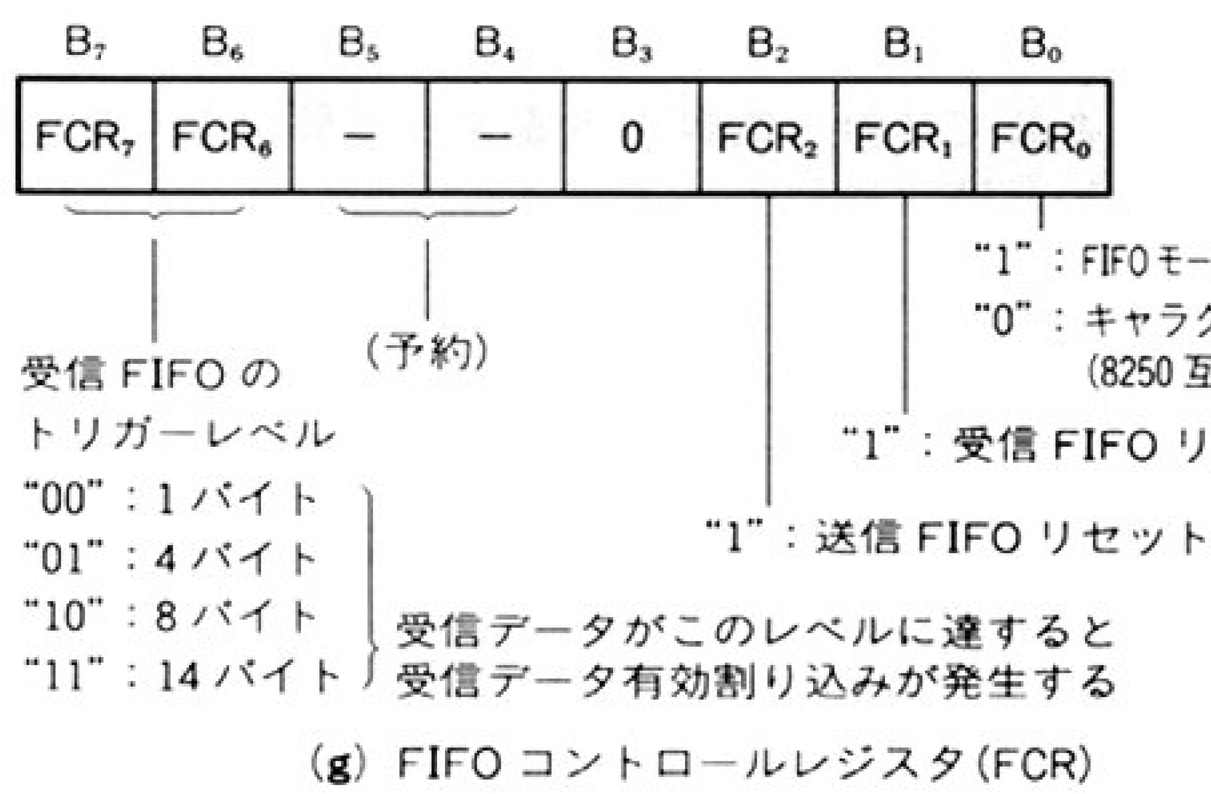
〈図2-1〉 16550A のレジスタ



DCTS, DDSR, TERI, DDCD は，対応する信号が前回参照された後に変化した場合 1 になります(デルタビット)．これらのどれかひとつでも 1 になると，モデムステータス割り込みが発生します．また，これらのビットは本レジスタを読み込むことでリセットされます．

● **FIFO コントロールレジスタ FCR**(Write only)
FIFO をコントロールします［**図2-1(g)**］．

● **ボーレート分周器ラッチ(LSB)DLL, (MSB)DLM**
(Read/Write)



〈表2-1〉 MC16550 IIの「割り込みレベル選択レジスタ」

ILRへの出力データ	割り込みレベル
00h	未選択
04h	INT0
05h	INT1
06h	INT2
07h	INT5

〈図2-2〉 RSB-384の「割り込みレベル情報取得レジスタ」ディップスイッチ(SW2)の状態を読み取る

B7	B6	B5	B4	B3	B2	B1	B0
0	0	〈4〉	〈5〉	〈6〉	〈1〉	〈2〉	〈3〉
		ポート3			ポート2		

← ディップスイッチ(SW2)の内容 1：ON，0：OFF

これらは合わせてLCRの **DLAB=1** のとき，端末通信速度を設定します．

DLLに端末速度分周比値のLSB(下位バイト)を，DLMにMSB(上位バイト)を出力します．

8251A系と異なり，クロックレート(ボーレートファクタ)は×16に固定されていますので，

設定する分周比値
＝基本クロック÷16÷設定したい端末速度

で与えられます．基本クロックは，MC16550 IIが7.3728 MHz，RSB-384が18.432 MHzです．

● 割り込みレベルレジスタ ILR

(Write only) MC16550 II

(Read only) RSB-384

割り込みレベルについてのレジスタですが，MC16550 IIとRSB-384とでは使い方がかなり異なります．

MC16550 IIでは「割り込みレベル選択レジスタ」として機能します(**表2-1**)．割り込みレベルはこのレジスタに出力するデータの内容で，ソフトウェア的に選択されることになります．

RSB-384では「割り込みレベル情報取得レジスタ」として機能します(**図2-2**)．割り込みレベルはボード上のディップスイッチ(SW2)でハードウェア的に選択されるので，必要に応じてその内容を読み取ることが

コラム●自動フロー制御機能が追加されたUART

16550Aは，16バイトの送/受信FIFOを内蔵しますが，自動フロー制御機能は持っていません．

したがって，フロー制御はドライバにおいてソフトウェア的に行わなければなりません．

Texas Instruments(TI)社で製品化された16550Aの上位互換IC TL16C550Cは，従来の仕様に加えて，それ自身にハードウェアフロー制御機能が追加されています〔TI社ではUARTとは言わずに，ACE(Asynchronous Communications Element)と称している〕．

具体的には，受信データがFIFO内に設定されたトリガーレベルに達すると，自動的にRTS信号をOFFにし，一方CTS信号がOFFに転じると直ちに送信を停止するというものです．

したがって，16C550Cどおしでのデータ転送ではこのフロー制御機能を利用するだけで十分で，ドライバではあえてフロー制御を考慮する必要がなくなり，とても心強い機能といえます．

この機能はMCRのB_5(AFE)を“1”にすることによりイネーブルされます．

先にも書いたとおり，16550Aではドライバでフロー制御を行わざるをえないわけですが，送信フロー制御については特に注意が必要です．

CTS信号を常時監視していて，これがOFFに転じたらドライバから16550Aの送信FIFOへのデータ送出を直ちに停止しますが，CTSがOFFになってもすでに送信FIFO内に送りこまれているデータの送信には「待った」をかけられません．

タイミングにもよりますが，最大で16バイトのデータが相手側の状態にかかわりなく送信される可能性があります．

相手側受信バッファがある程度の大きさを持っていて，たとえ受信フローOFF状態でも16バイト程度の受信を許容できるのなら，それでも一向に差し支えありません．

しかし16550Aと16C550Cとを対向させてデータ転送を行う場合，16C550C側ではそれ自身のオートフロー機能だけに依存するわけにはいきません．

受信フローが働いてRTS信号をOFFにしても，相手側16550Aはデータの送信を直ちには停止してくれない可能性が大きいからです．その場合，16C550C側でのオーバーランエラーによるデータの取りこぼしは避けられません．

最近，このTL16C550Cを採用したシリアルインターフェース・ボードが製品化されています〔SRS-98(ヒューマンデータ)，RSA-98(アイ・オー・データ機器)〕が，これらのボードを用いる場合には上記のような注意が必要です．

このレジスタの目的となります．

I/O ベースアドレス

MC16550 IIでは，3ビットのディップスイッチでアドレスの最下位4ビットを **xxx0h** から **xxxEh** まで(偶数番地のみ8通り)選択可能です．

一方，RSB-384は8ビットのディップスイッチ(SW1)でアドレスの上位8ビットを **00h** から **FFh** までフルにデコードできます．

MC16550 IIとRSB-384のレジスタとI/Oアドレスの関係を**表2-2**に示します．

割り込みレベル

割り込みレベルは前述したように，MC16550 IIではプログラムからILRにデータを出力することによりソフトウェア的に選択されます．

この場合，はじめから二つのポートは同じ割り込みレベルになり，OR条件で割り込みが働きます．逆にいうと，MC16550 IIでは二つのポートの割り込みレベルを独立させて働かせることはできません．

一方RSB-384では対照的に，割り込みレベルはボード上のディップスイッチでハードウェア的に，しかも二つのポートについてそれぞれ別個に設定する仕様になっており，二つのポートの割り込みレベルを同じに設定することはできません．

PC98では，ユーザーに開放されている割り込みレベルがもともと少ない上，フロッピディスク・ドライブやハードディスク・ドライブのインターフェースにそれらを割かなければなりません．二つくらいのポートであれば同一割り込みレベルで駆動することによって，割り込みレベルの浪費？を避けたいところです(マルチチャネルのホストシステムとなるとなおさら…)．

そういう面ではMC16550 IIの仕様が好ましいわけですが，RSB-384でもボードの比較的簡単な改造によって同様な使い方をすることができます(第1回のコラム参照)．これ以降，連載で扱うRSB-384については改造を施されたボード(二つのポートを同一割り込みレベルで駆動可能)であることを前提とします．

〈表2-2〉 MC16550 IIとRSB-384のレジスタとI/Oアドレスの関係

	MC16550 II		RSB-384	
	ポート2	ポート3	ポート2	ポート3
RBR	00Dxh	08Dxh	xxB0h	xxB1h
THR	00Dxh	08Dxh	xxB0h	xxB1h
IER	01Dxh	09Dxh	xxB2h	xxB3h
IIR	02Dxh	0ADxh	xxB4h	xxB5h
FCR	02Dxh	0ADxh	xxB4h	xxB5h
LCR	03Dxh	0BDxh	xxB6h	xxB7h
MCR	04Dxh	0CDxh	xxB8h	xxB9h
LSR	05Dxh	0DDxh	xxBAh	xxBBh
MSR	06Dxh	0EDxh	xxBCh	xxBDh
DLL	00Dxh	08Dxh	xxB0h	xxB1h
DLM	01Dxh	09Dxh	xxB2h	xxB3h
ILR	10Dxh(共通)		xxBFh(共通)	

xとxx部分は，ディップスイッチにより決まる

コラム●できることなら「非」割り込み駆動のドライバにしたい

今回解説しているドライバは，ボードの仕様上ハードウェア割り込み駆動にせざるをえません．

では，なぜ割り込み駆動にしなければいけないのか考えてみましょう．

データ通信の世界では，受信データを取りこぼしてはいけないということが絶対的な命題ですから，その時ホストCPUがたとえ何をしていたとしてもそれを一時中断し，最優先で受信データを取り込まなければいけません．

また，「受信データを絶対に取りこぼさない」という目的のためには，ボード自体にフロー制御機能を持たない8251A系や16550A系のボードでは，ソフトウェア的にフロー制御を行わなければならず，ドライバにおける割り込み処理は必須でした．

ところがシリアルボードで最近目立つのは，ある程度のサイズのバッファを内蔵し，それ自身でオートフロー制御が可能という，インテリジェント化されたボードで，ドライバを割り込み駆動にする必要もなくなります．

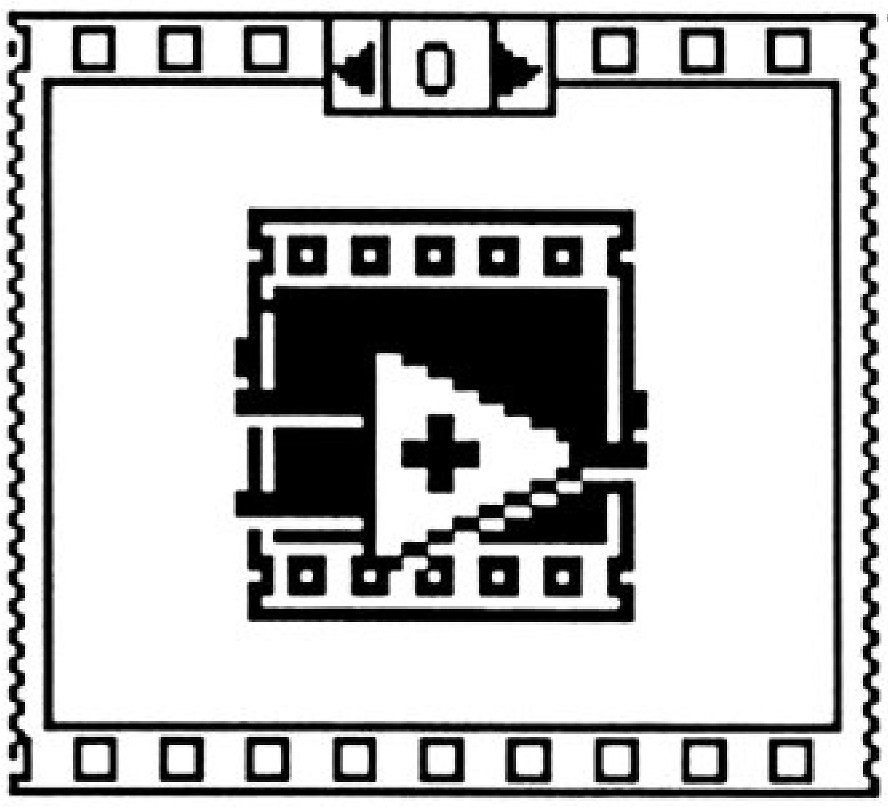
これらのボードでは，フロー制御をきちんと動作させれば受信データの取りこぼしは起こらないはずです．

これによりどんなメリットがあるかというと…

(1) ドライバのプログラミングが省力化される．
(2) ドライバのコードサイズが激減する．
(3) ドライバ自身のバッファはいらないから，常駐サイズはさらに縮小する．
(4) ホストCPUのオーバーヘッドが減少する．
(5) ただでさえ少ない拡張スロットの割り込みレベルを浪費しないですむ．

いいことずくめですね．

連載 第3回



計測制御のためのツール LabVIEW プログラミング

LabVIEW で小品をつくってみよう！

計測マニア

今回は，いくつかの小さなプログラムを作っていきたいと思います．

まず，電光掲示板の vi を最初に作り，そして RS-232C ポートと電子天秤を接続して，重さを測定するドライバ vi を作ってみたいと思います．最後に，その vi を使ってデータをテキストファイルへ書き出すアプリケーションを作ります．

電光掲示板をつくってみよう

● 仕様を考える

では，最初の課題として電光掲示板を作ってみましょう．仕様をどんなものにするか考えましょうか．まず，表示文字の速度を好みに合わせて 3 段階ぐらいに変えられるものを作って見ましょう．

絵も表示したいという気持ちになるのですが，とりあえず今回は，文字列を 6 文字ずつ右から左へ流しながら表示するプログラムを考えてみたいと思います．完成後の姿は，**図3-1** のようになります．

● パネルレイアウトをデザインする．

最初は，表面に見えるパネルをデザインしましょう．まず，File メニューから「NEW」を選んで新しいパネルを準備します．そして，Controls メニューから文字列表示器を選びます．これには，名前を「表示」と入力してください．

次に，大きく表示するためにマウスで矢印ポインタを押して(実行ボタンではありませんよ)，先ほど作った表示器の右下へポインタを移動させます．

すると，ポインタが L 字を横反転した形に変わり拡大可能な状態になるので，マウスを押し下げたまま好みの大きさになるまで大きくします．この様子を表現したのが**図3-2** です．

次に，**図3-3** に見られるアイテムを配置します．**図3-3** には，LabVIEW のプルダウンメニューからどのアイテムを選んだのかアイコンで表示してみましたので，これを参考に配置してみてください．

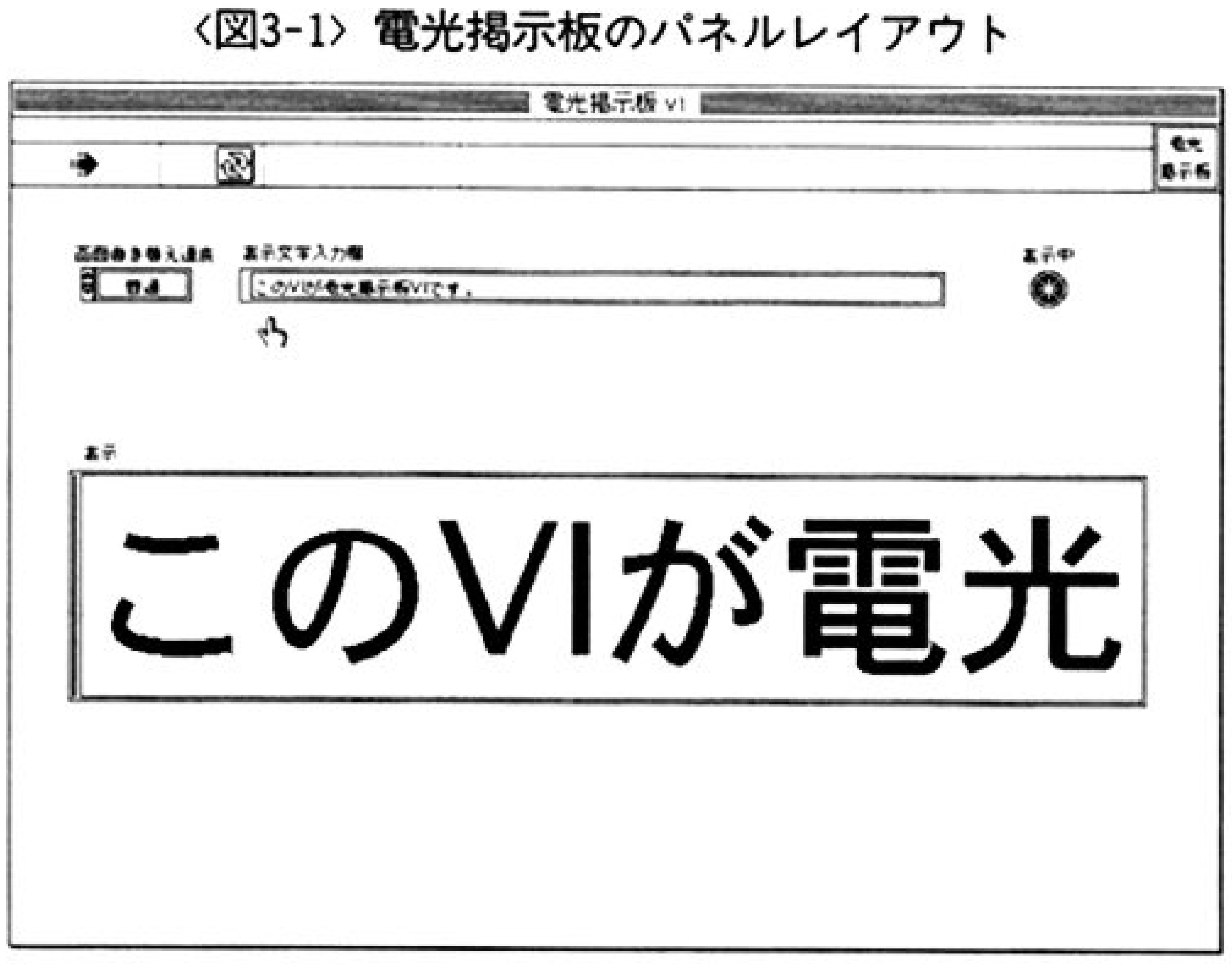
そして，今度は表示速度を変えるために「表示速度切り替え」に「おそい」，「ふつう」，「はやい」などの文字を追加します．**図3-4** に示しましたので，これを参考にしてアイテムの追加をしてみてください．

さて，最後に表示器の表示文字を多くするために，**図3-5** の手順で文字の大きさを 36 ポイントに変えてみます．

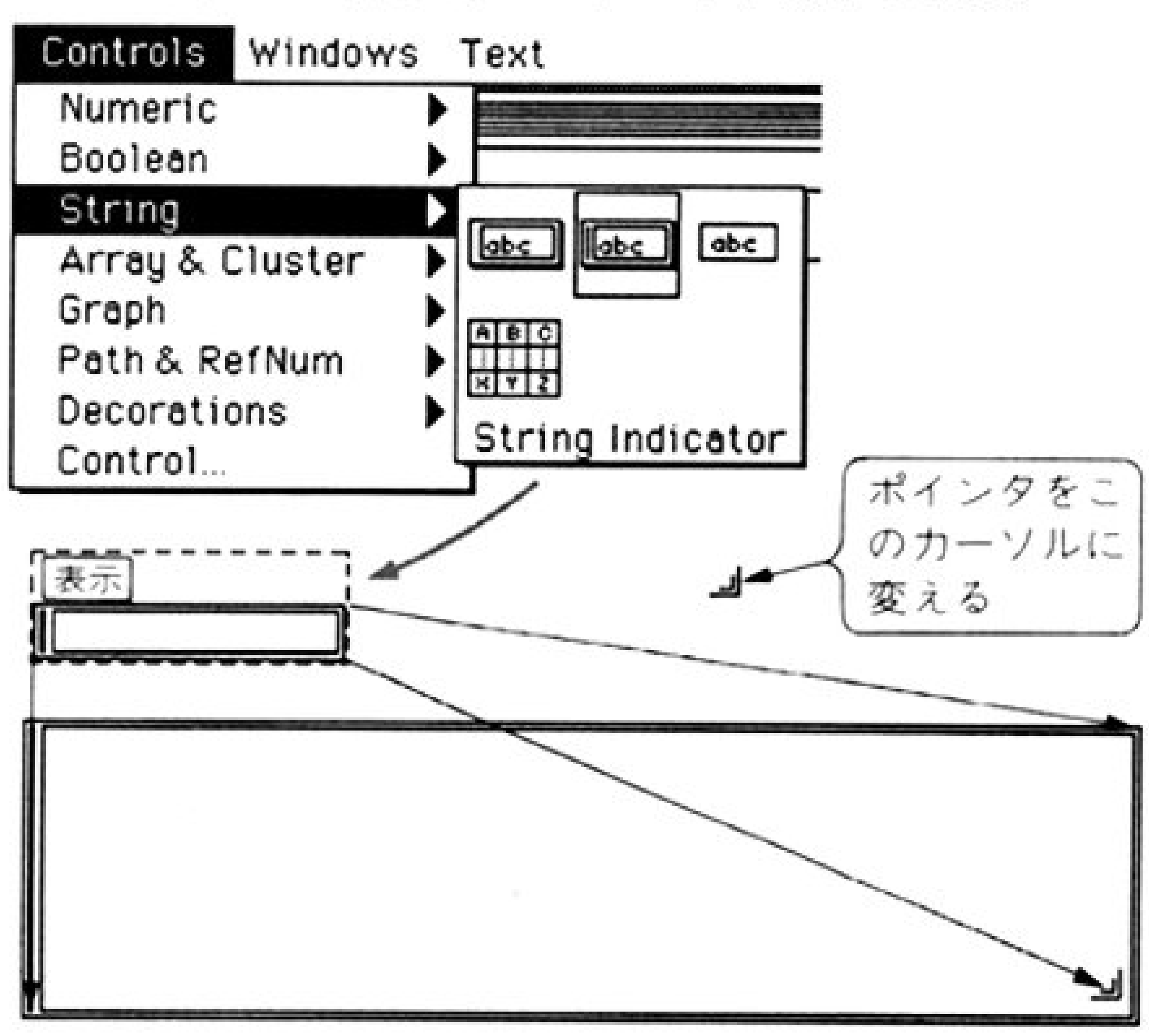
● ダイアグラムの中身を作る

これで，表面に見えるパネルはできたので，次はダイヤグラムの中身を作ります．制御構造は，連続実行なので **While** ループを使うことにします．

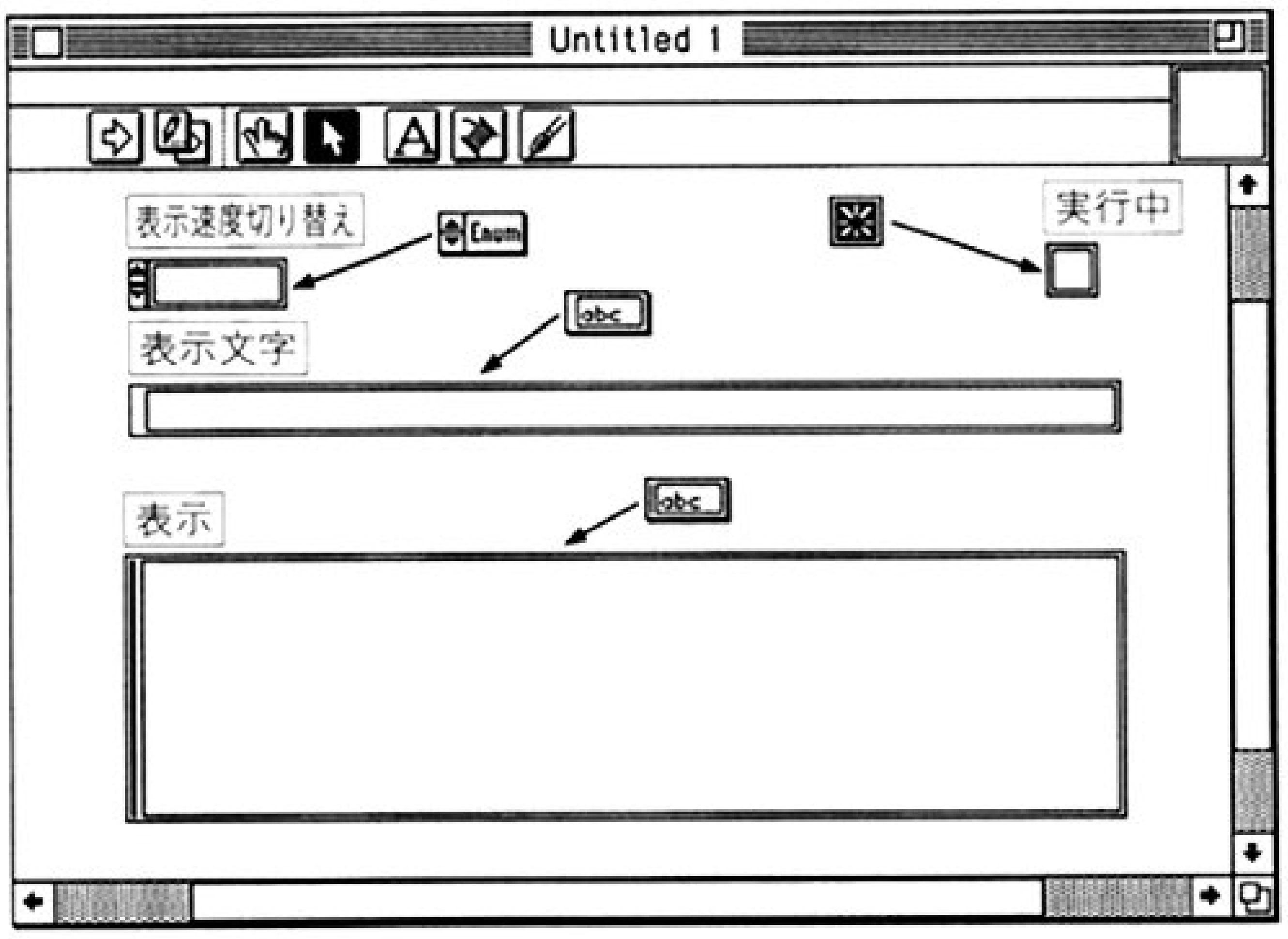
〈図3-1〉 電光掲示板のパネルレイアウト



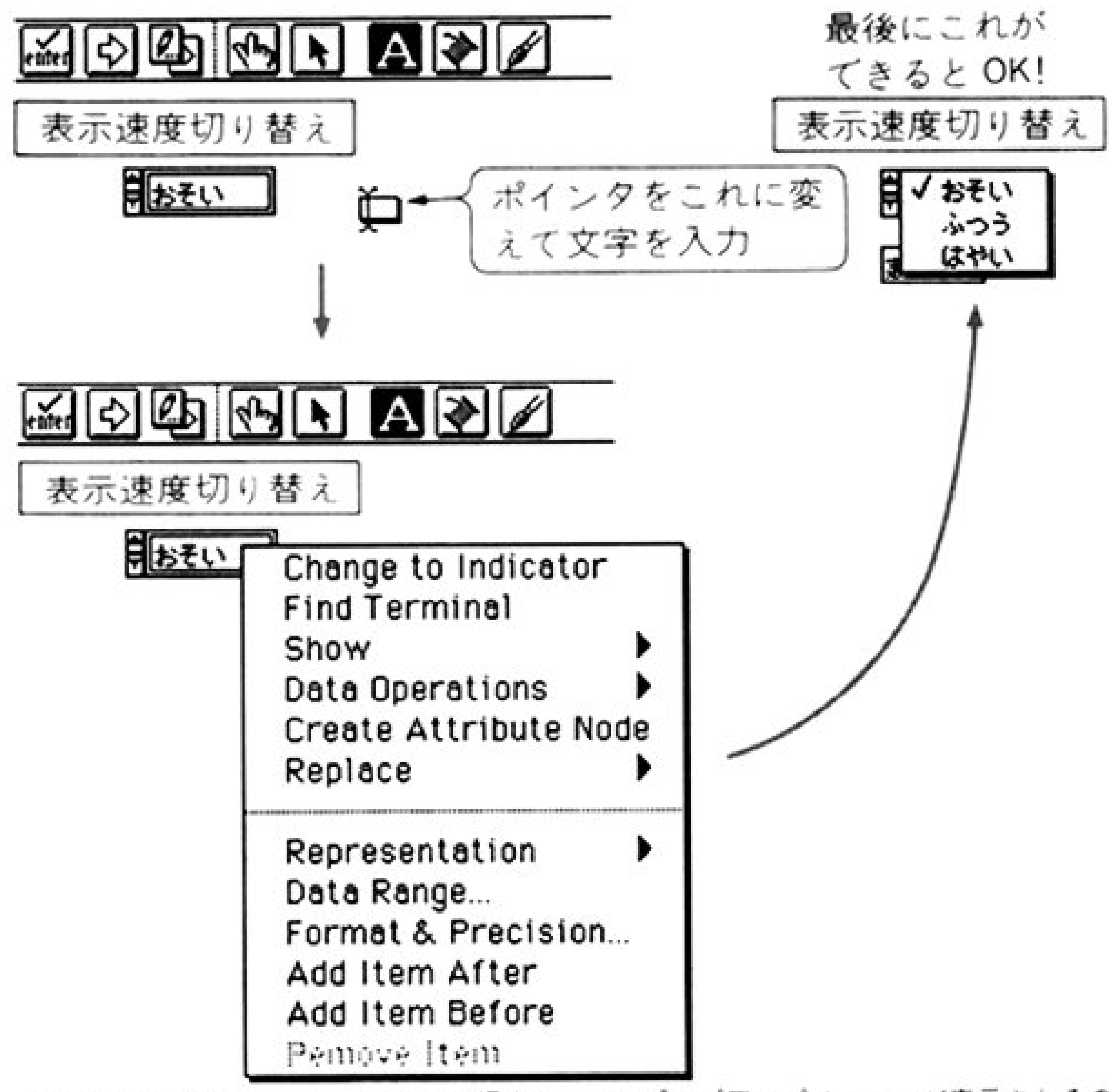
〈図3-2〉 表示用のパネルを準備する手順



〈図3-3〉 必要なアイテムを配置する

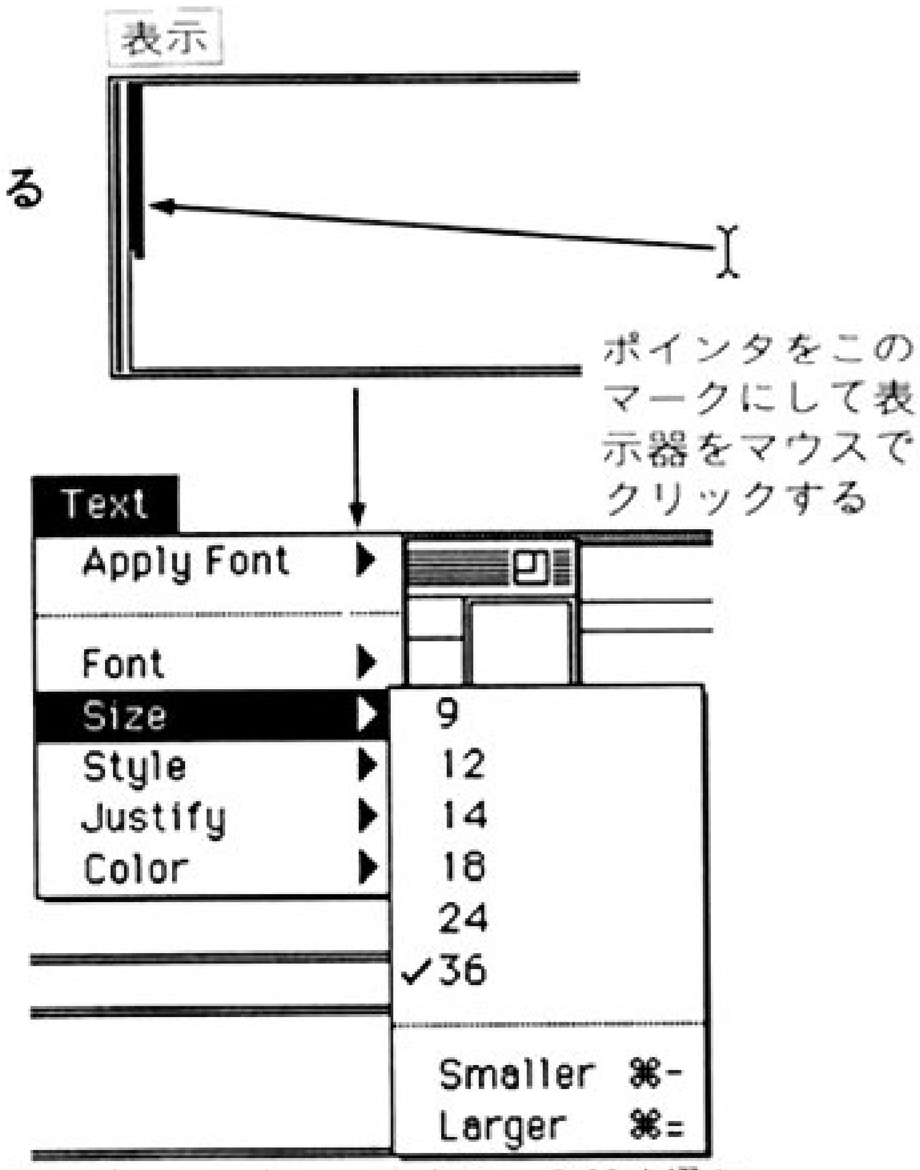


〈図3-4〉 表示速度用の文字を準備



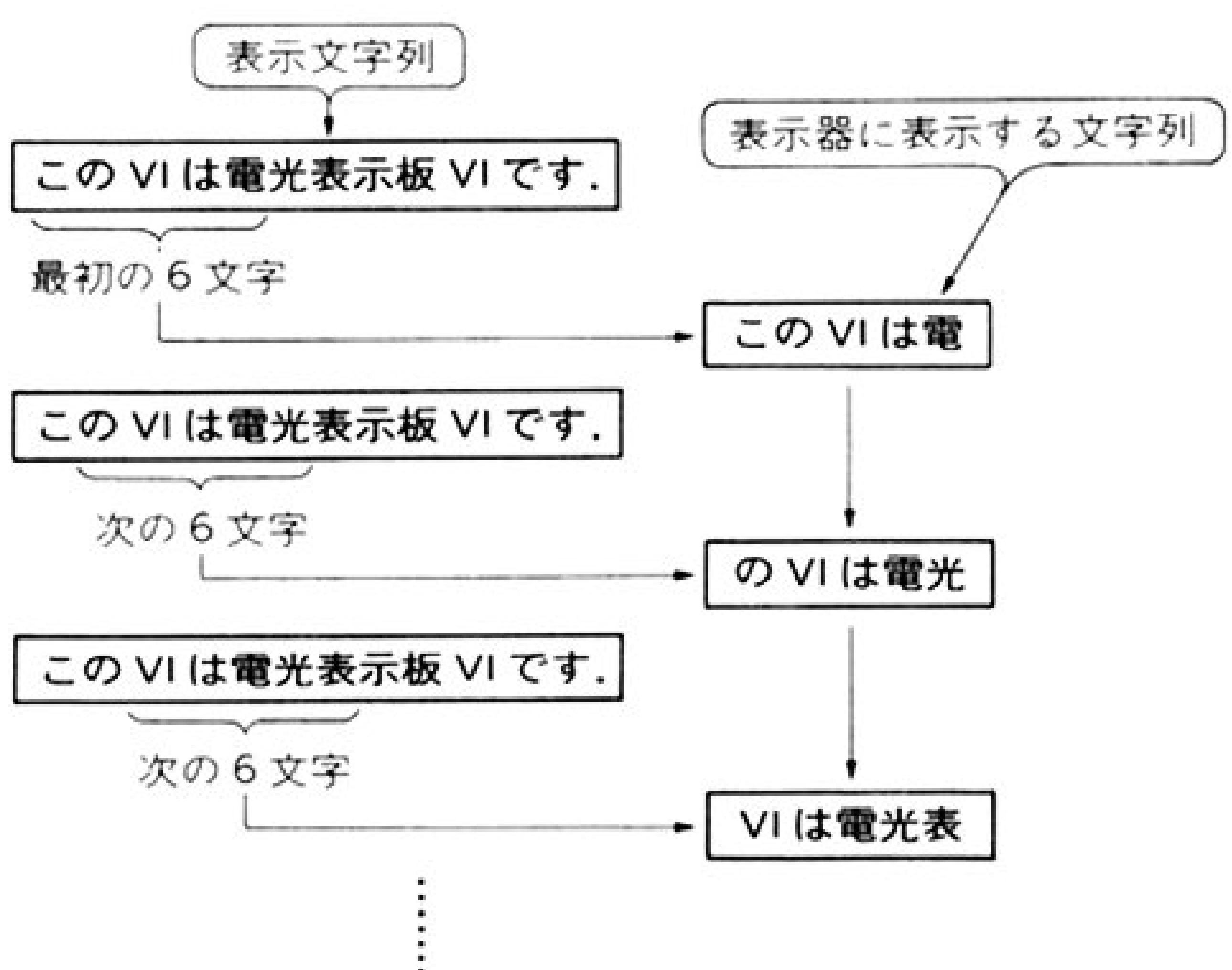
コマンドキーを押しながらマウスを押すと，このポップアップメニューが表示されるので，Add Item After を選んでポップアップの選択肢を追加する．

〈図3-5〉
文字を大きくする



そして，Text メニューから Size の 36 を選ぶ

〈図3-6〉 文字列を移動させるアルゴリズム



そして，文字列を 6 文字ごとに順番に表示して行くことにより，あたかも文字列が右から左へ流れていくように表示することにします．このアルゴリズムを図で表現すると**図3-6** のようになります．

そこでまず，**図3-7** に示したように **While** ループを置いて拡大し，適当にオブジェクトをループの中に入れます．

次に，文字を前述のアルゴリズムで 6 文字ずつ取り出して表示することを考えなければなりません．今回はこの目的のために，「String Subset」という vi を使ってみます．

この vi は，Functions メニューの「String」にあります．機能を説明したのが**図3-8** ですが，うってつけの vi です．

次に，文字の表示速度を三つの速度に変える機能を考えます．この目的のために，最初 **While** ループを置きます．次に，ループの実行速度を変えるために，ループの中に **Wait Until Next ms Mulitple** を置きます．

こうすることにより，決まった時間間隔ごとにループを実行させることが可能になります．また，速度を三つに変える必要があるので，これにケースを置いて時間を三つ選択して出力するようにダイアグラムを変更します．これらの例が，**図3-9** です．

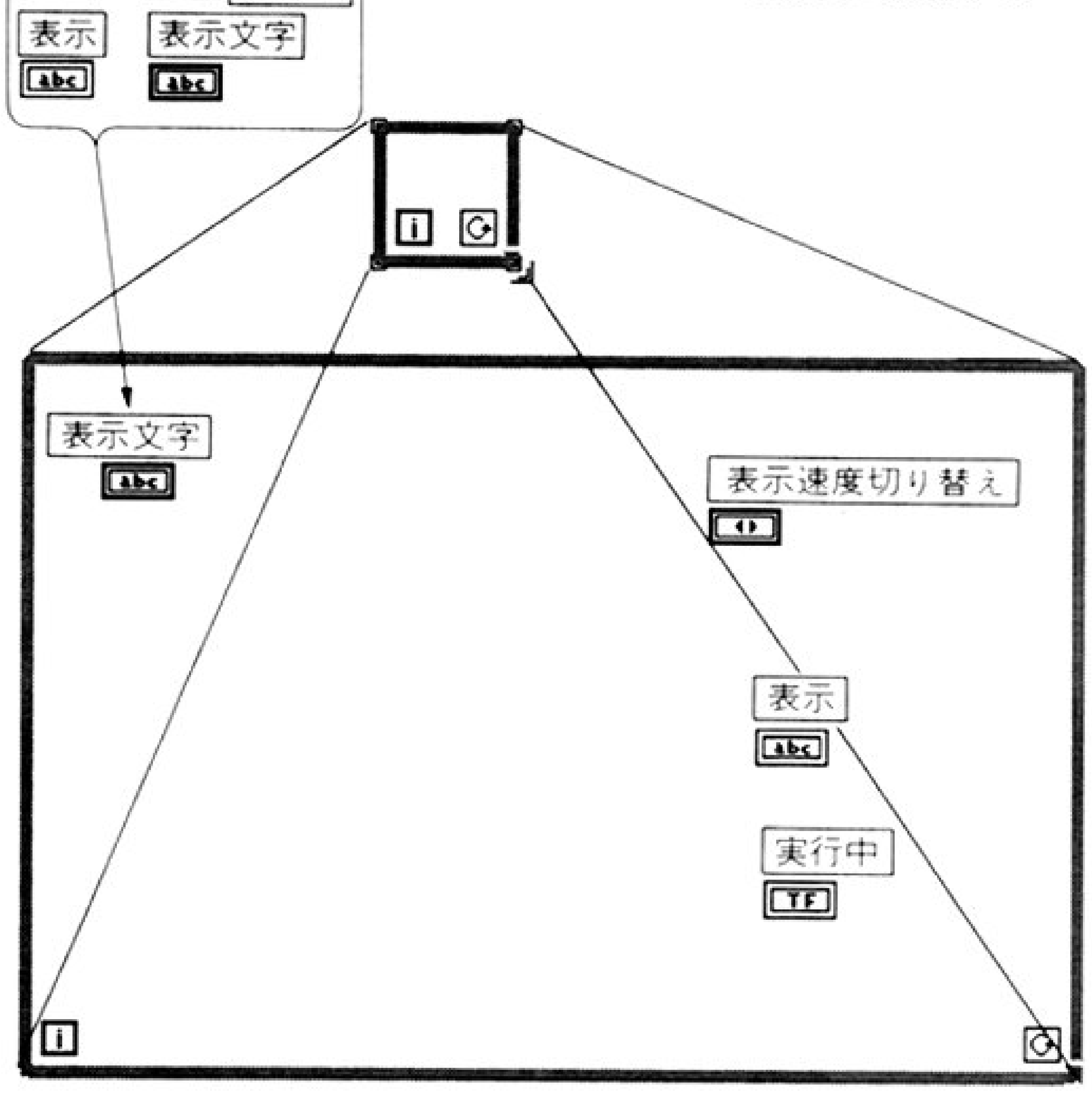
その他，最初に文字列全体のバイト数を数えて，ループを何回まわすかを決定するために，文字列の長さを数えるファンクションや空白文字列を表示するために，6 文字分の空白文字列を先頭に連結するファンクションを使います．その例が，**図3-10** です．

これまでに述べたファンクションを組み合わせて完成したダイアグラムを，**図3-11** に示します．**図3-12** はそれを解説したものです．

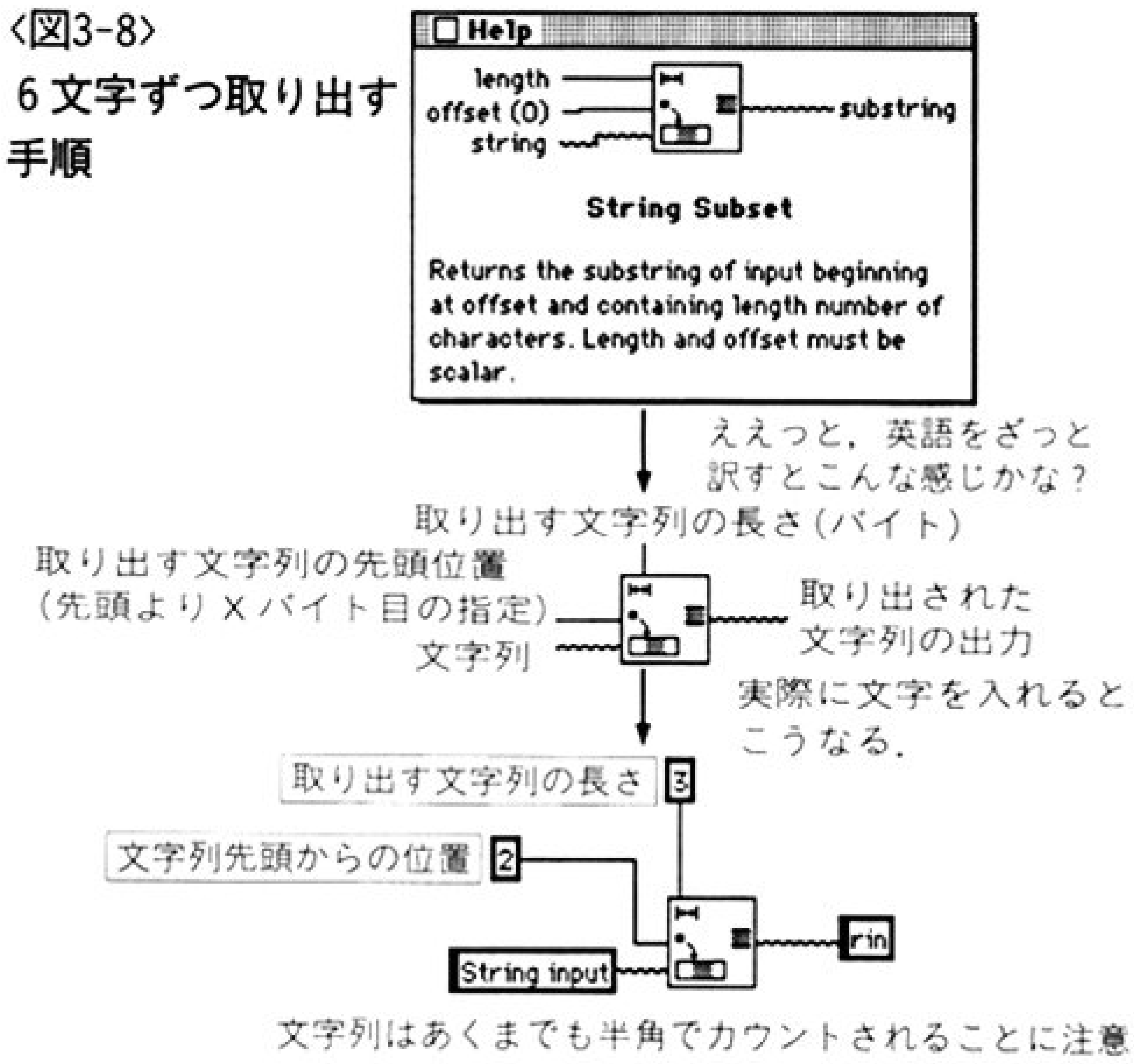
● 電光掲示板 vi の完成

さて，以上で動作に関する部分は完成しましたので実行してみましょう．正確に，**図3-1** と同じにするためにはアイコンをつくったり，ウィンドウタイプを変えたりしなければなりませんが，とりあえずここではパ

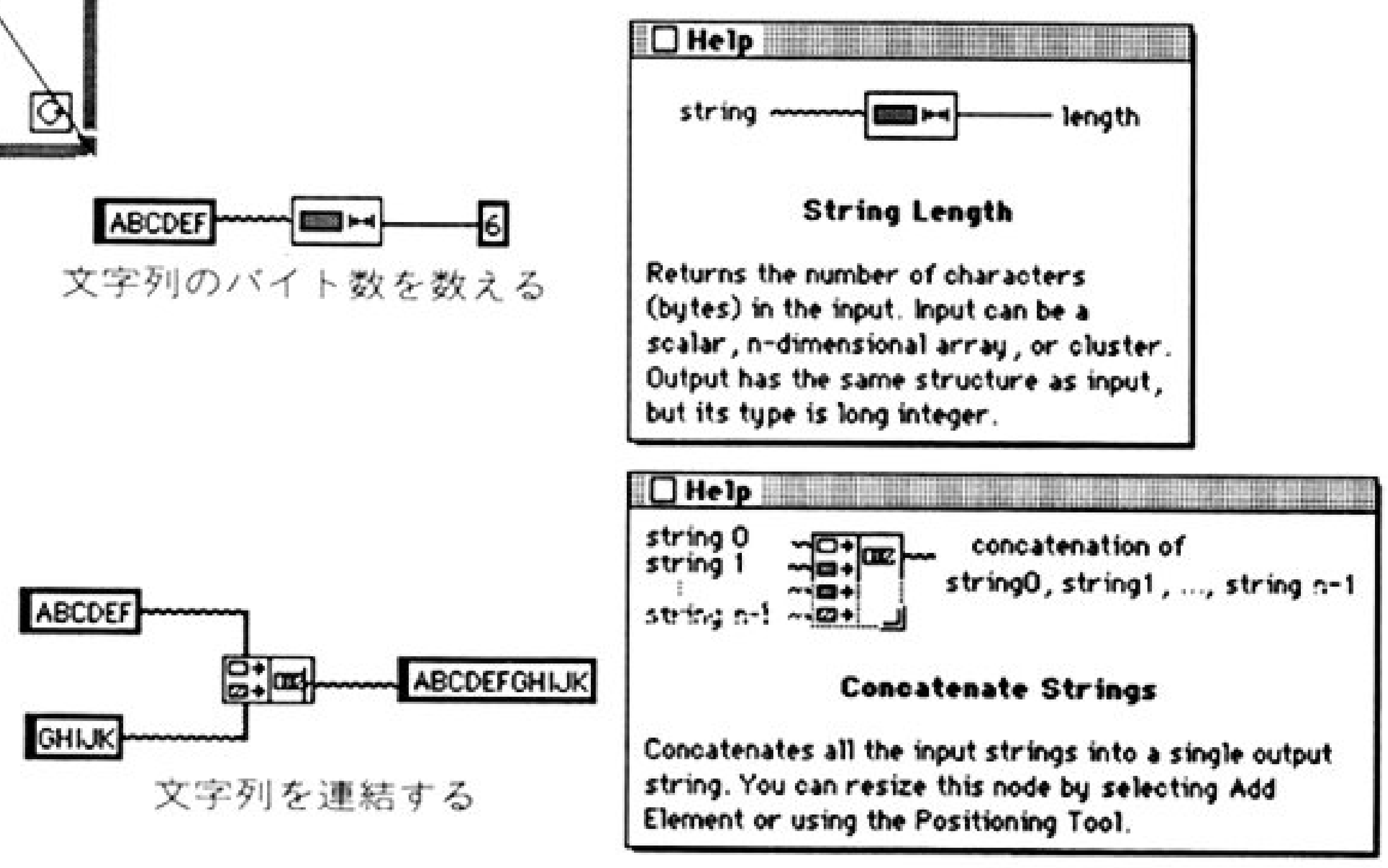
〈図3-7〉 while ループで順番に表示する



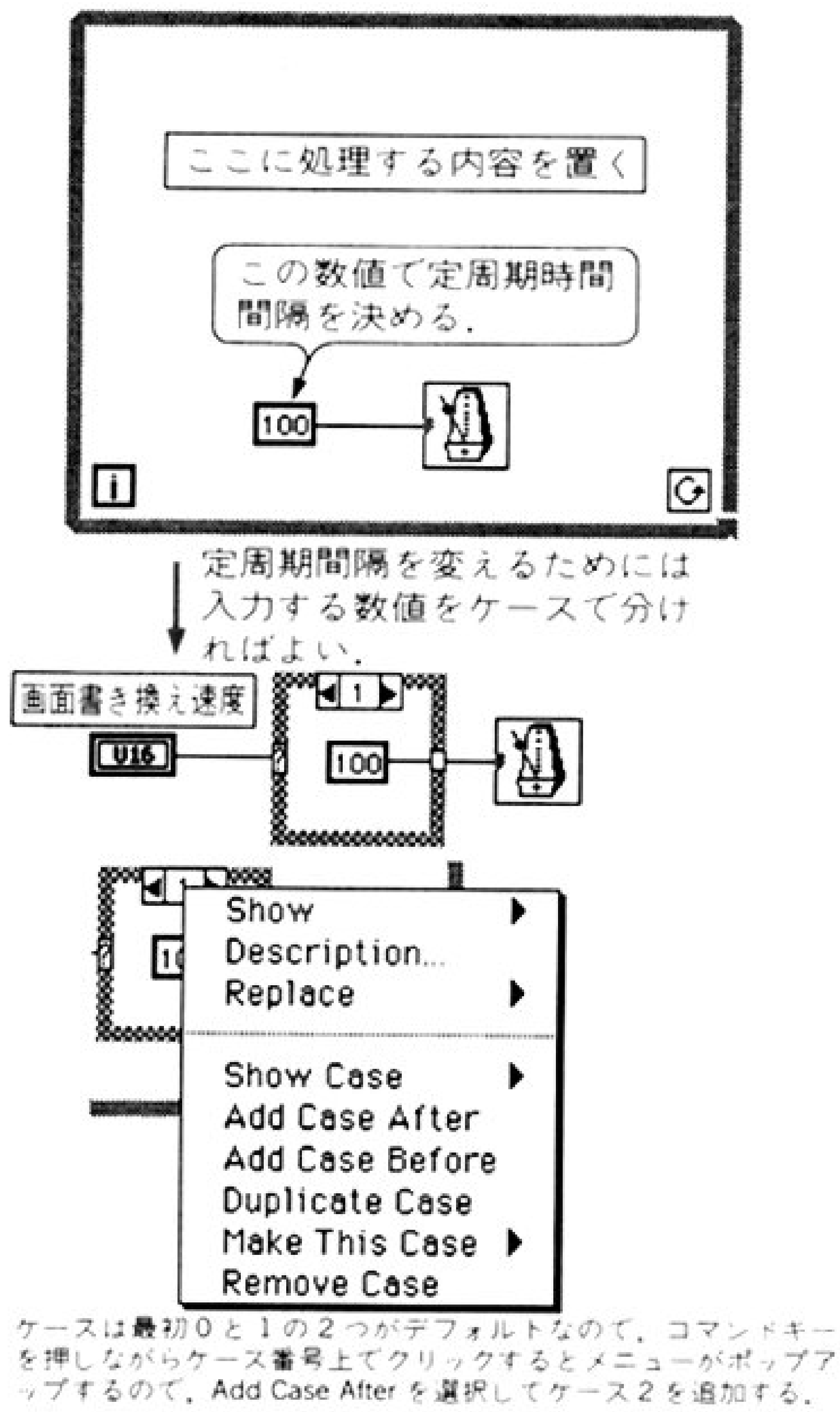
〈図3-8〉 6文字ずつ取り出す手順



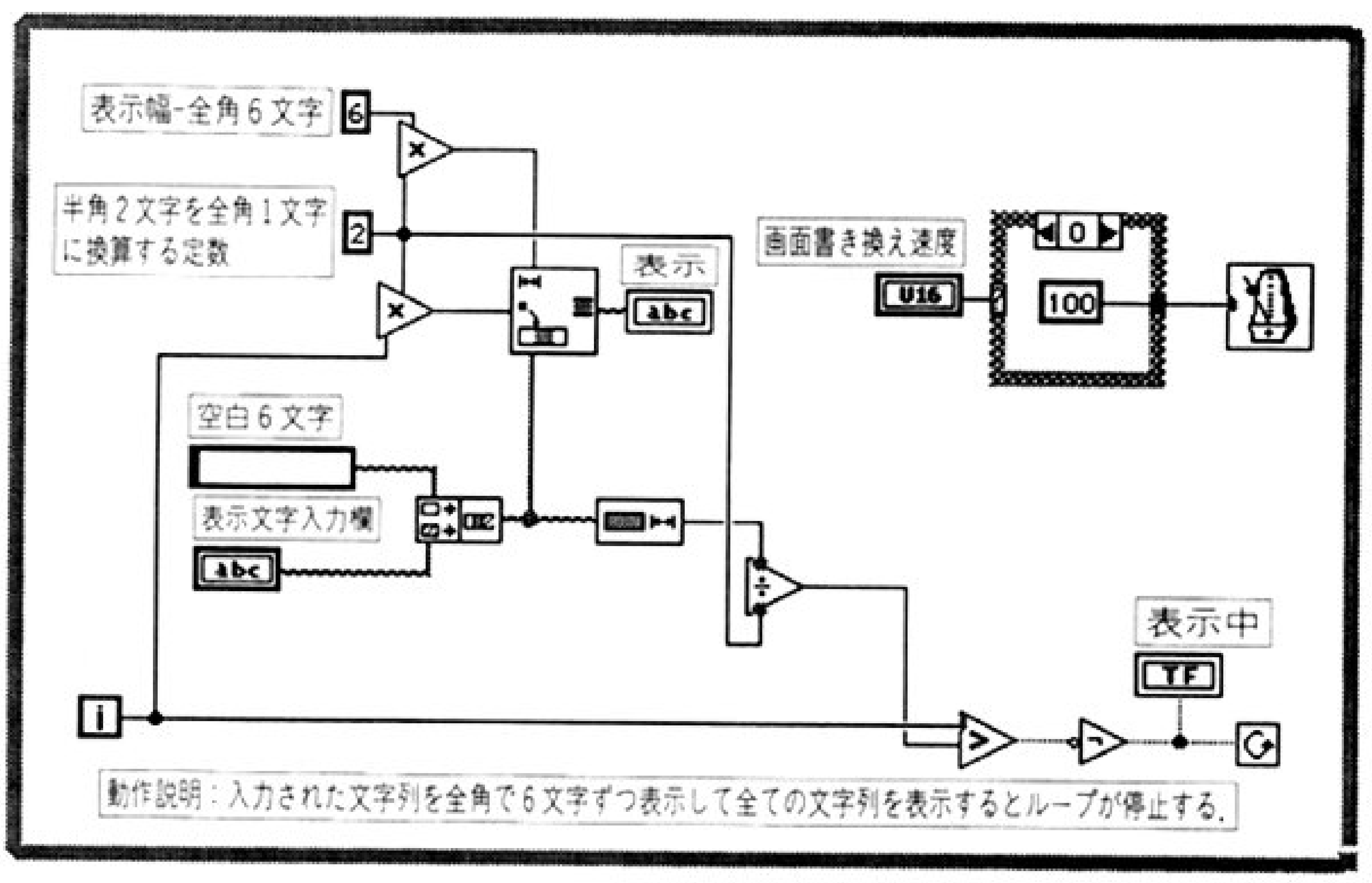
〈図3-10〉 ループを何回まわすかを決める



〈図3-9〉 速度を三つに変える



〈図3-11〉 完成したダイヤグラム



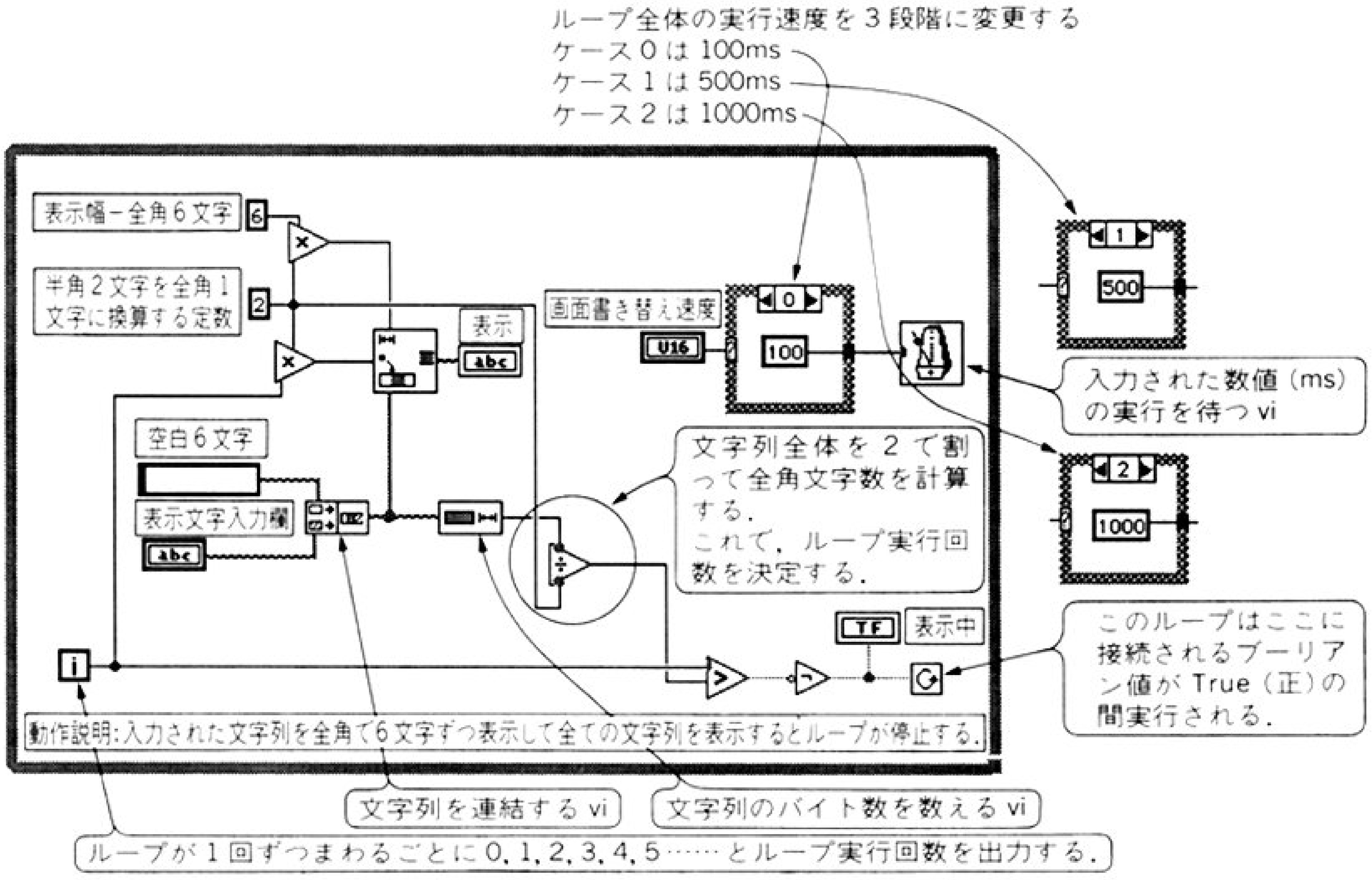
スします．

では，実行ボタンを押してみてください．うまく行けば，**図3-13** のように実行されるはずです．うまく行かなかった方は，前回に説明したエラー表示機能を使ったり，デバッグランプを使って実行状況をモニターしてデバッグしてみてください．

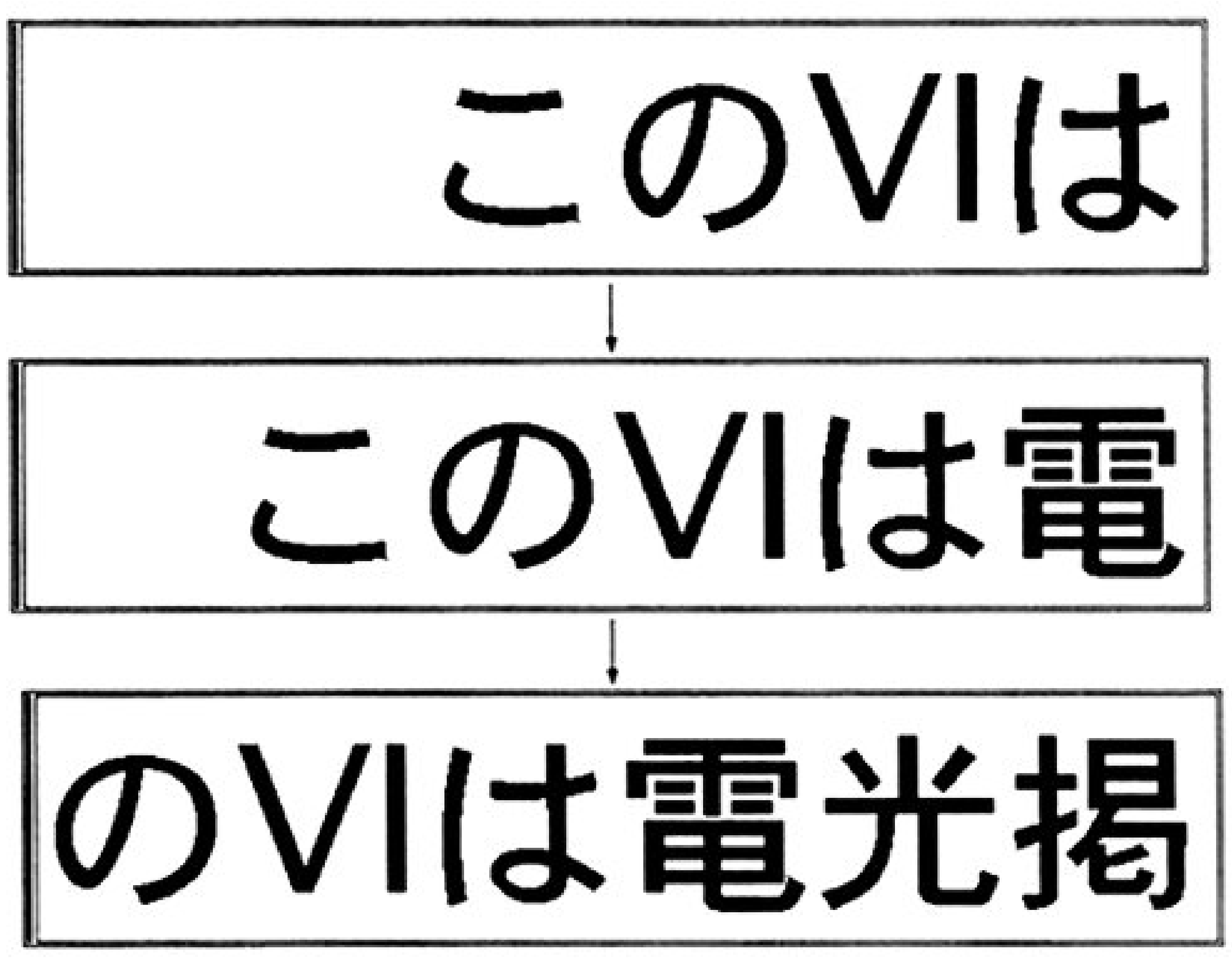
シリアルドライバをつくってみよう

では次に，RS-232C ポート付きの電子天秤を制御する vi ドライバを作ってみたいと思います(**図3-14** 参照)．今回取り上げたのは，㈱エー・アンド・ディの電

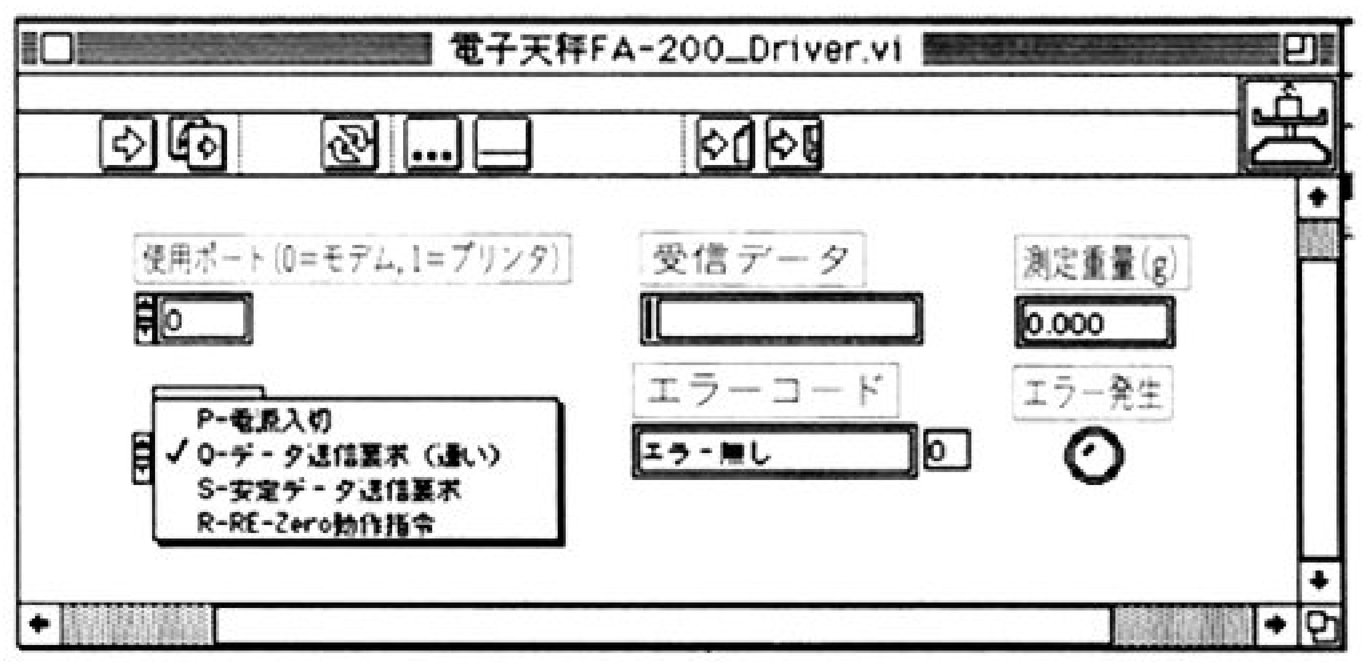
〈図3-12〉
完成したダイヤグラムの内容



〈図3-13〉 完成した電子掲示板の実行例



〈図3-14〉 電子天秤を実行するviドライバ



子天秤 型式FA-200です．

● **電子天秤の仕様を調べる**

この製品の取扱説明書によれば，次のような仕様になっています．かなり精密な天秤であることが分かります．RS-232Cポートはオプションとなっています．

機種：FA-200
最大秤量：210 g
最小表示：0.001 g
再現性(標準偏差)：0.001 g
直線性：±0.002 g
感度ドリフト(10℃～30℃)：±3 ppm/℃
安定所要時間：約2.5秒(代表値)

次に，この電子天秤を制御するためには，RS-232Cのコネクタ仕様と伝送の仕様を知る必要があります．RS-232Cのコネクタは，一般的なDサブ25 Pinのメスコネクタを使っています．製品の取扱説明書によれば，**表3-1**のような仕様となっています．

次に知らなければならないことは，伝送速度やパリティなどの通信仕様です．これも取扱説明書によれば以下の設定です．

方式　　：EIA　RS-232C
伝送方式：調歩同期(非同期)式
信号形式：ボーレート600/1200/2400/4800 bps切り替え可
▶データビット長：7ビット
▶パリティ　　　：1ビット(EVEN)
▶ストップビット：1ビット
▶使用コード　　：ASCII

〈表3-1〉 電子天秤のRS-232Cコネクタ

ピンNo.	信号	天秤から見た信号を方向	意　味
1	FG	－	フレームグラウンド
2	RxD	入力	受信データ
3	TxD	出力	送信データ
4	RTS	入力	送信要求
5	CTS	出力	送信許可
6	DSR	出力	データセット・レディ
7	SG	－	シグナルグラウンド
8～25	NC	－	NON-CONNECTION

〈表3-2〉電子天秤FA-200のデータフォーマット

状　態	表示例	送信データ ヘッダ，データ，単位，ターミネータ
安定時	○200.000 g	ST，+0200,000 _ _g CR LF
不安定・マイナス	−1.25 g	US，−00001.25 _ _g CR LF
オーバーロード	E	OL，+9999999E+19 CR LF
オーバーロード	−E	OL，−9999999E+19 CR LF

〈表3-3〉Macと電子天秤を接続するケーブル

Mac側	電子天秤側
DIN 8 Pin オス	D サブ 25 Pin オス
4(SG)	7(SG)
3(RCV)	2(RxC)
5〔XMIT(−)〕	3(TxD)

そして，さらにこの電子天秤に与えるコマンドを知る必要があります．コマンドは次のとおりです．

コマンドには，次の4種類があります．なお，< >はターミネータです．設定したターミネータを付加してください．

P<term>：Power ON/OFF

ON/OFFキーを押したのと同じ動作をします．

Q<term>：send Quickly

安定・不安定にかかわらず，1データ返信します．

S<term>：send Stable data

安定時の1データを出力し，表示を1回ブランクします．キー・BモードにおいてPRINTキーを押したのと同じ動作をします．

R<term>：Re-zero

RE-ZERO動作を行います．

これで終わりではありません．次に必要なことは，データフォーマットを知ることです．これも取扱説明書によれば，**表3-2**のような内容となっています．

▶ヘッダは，次の3種類．

ST……安定(表示は安定している)

US……不安定(表示は変動中である)

OL……オーバーロード(E，−E)

▶データはASCIIの数字で，次の文字・符号を含む．

+(**2BH**)，−(**2DH**)，・(**2EH**)，E(**45H**)

▶単位は次のとおり．_はスペース(**20H**)を示す．

単位	送信データ
g	_ _g

▶送信データのターミネータは常に <CR> <LF>．

これで，やっとドライバviを作るために必要なデータがそろいました．

最後に，電子天秤の設定を以下のとおりにしてください．FA-200の設定は，次のとおりです．

C1-1，C2-1，C3-3，C4-3，C5-1，C6-0

この設定は，表示オフの状態で次の手順によって変更することができます．

(1) RE-ZEROキーを押しながらON，OFFキーを押す．

(2) MODEキーを押すと現在の設定が表示される．

(3) ここで設定を変えたいときはRE-ZEROキーによって設定が変わる．

(4) MODEキーで次の項目へ変わる．

(5) 途中で抜けるときはON，OFFキーを押す．

● 接続ケーブルを作る

MacintoshにRS-232C機器を初めて接続するときに，最も悩むのがこの問題であるような気がします．MacintoshはRS-422規格であるがゆえに，取扱説明書にあるRS-232Cの説明が全く役に立ちません．

では，どうしたらよいでしょうか．いろいろ調べてみると次のようなことが分かりました．

▶ RS-232CとRS-422を接続するには，シグナルグラウンドをまずつなぐ必要がある．また，RS-422は平衡信号だが不平衡の使い方もできる．

▶ XON，XOFFのソフトハンドシェイクを用いたフロー制御であれば，RS-232C側はRTSとCTSをジャンパして，あたかもMac側(RS-422側)が送信可能であるように条件成立させればよい．

その他の文献も参考にして，今回は**表3-3**のような結線のケーブルを作りました．RS-232C側は，コネクタ側で4番ピン(RTS)と5番ピン(CTS)をジャンパしてください．この結線に関しては，参考文献(5)が非常に参考になります．

その他，ささいなことですが，Mac側のDINコネクタはパーツで個々に購入せず，Mac用にモールド済みのケーブルを購入して反対側を切り落とし，次にRS-232Cコネクタを接続する方法がお奨めです(例えば，Mac用プリンタケーブルを購入して加工する)．

というのも，DINコネクタは端子が非常に密集していてハンダ付けが非常に困難であり，またハンダ付けのときの熱で端子の位置がずれて接触不良が発生しやすいからです．

一度，結線をテスターで何回も確認してOKだと思っていたケーブルが，時々接続エラーになるので調べてみると，接触不良を起こしていたことがありました．以降，DINコネクタは個々に購入していません．

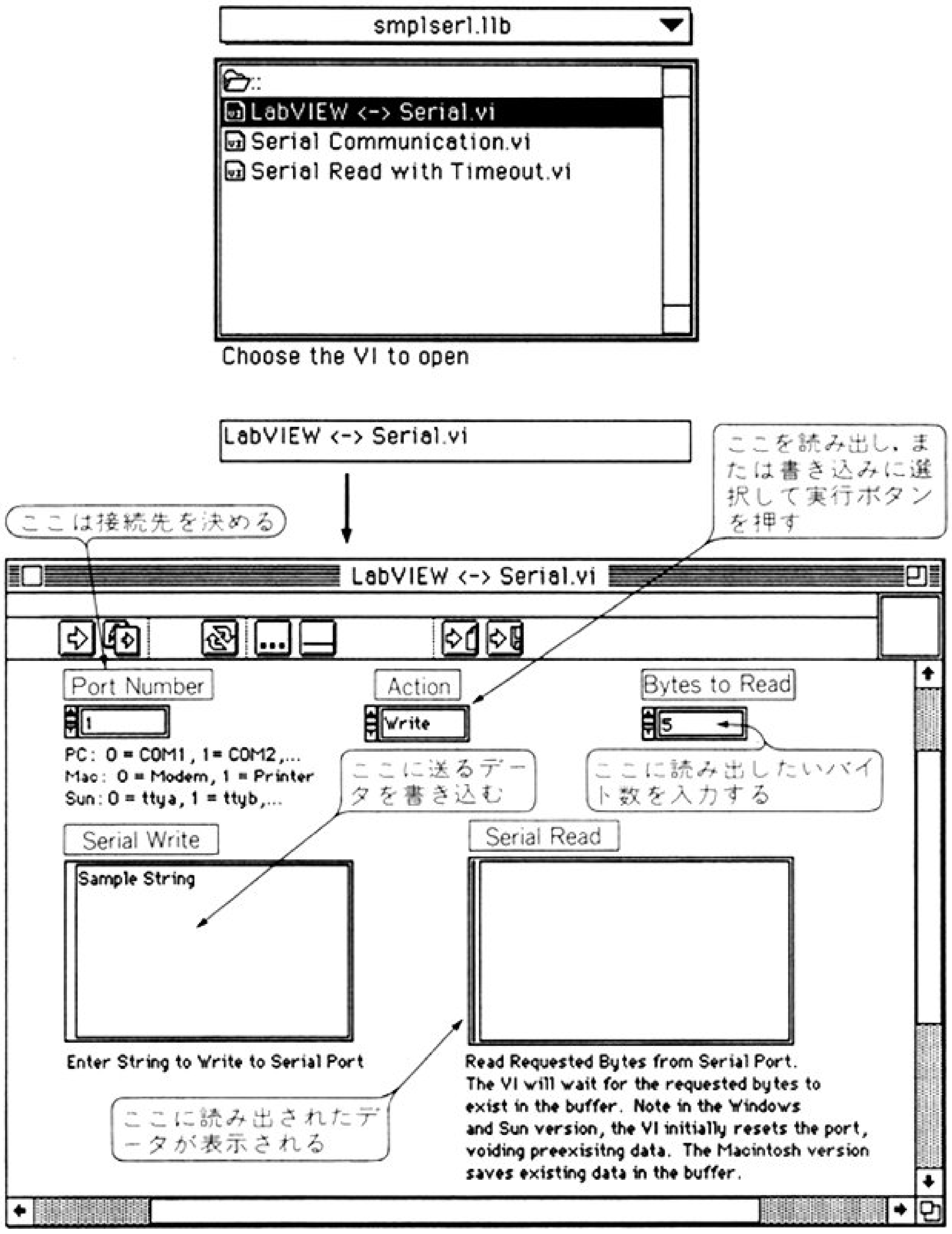
● 実験してみる

では，ケーブルもできたことなので，さっそく接続できるか，コマンド，通信条件が取扱説明書どおりに一致するか，取扱説明書に書いていない応答や異常応答がないか，などを実験してみます．

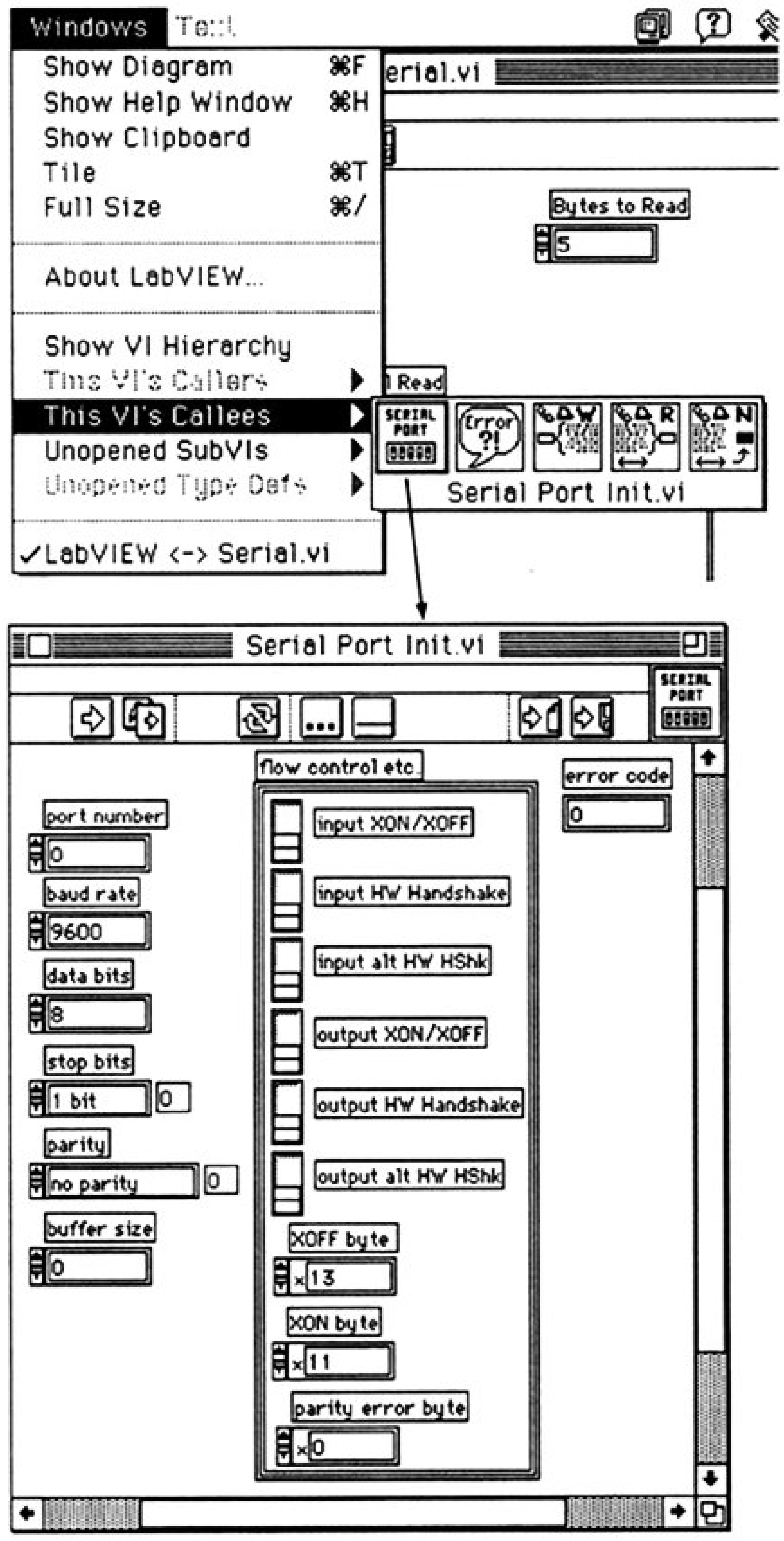
今回実験してみると，コマンド表ではオーバーロードのときはエラーデータを返すはずですが，プロトコルアナライザを使って調べてみると，何もエラーを返してこない現象が見られました．

この実験のために，LabVIEWで準備されているvi

〈図3-15〉Serial.vi の画面



〈図3-16〉通信条件を設定する



は Examples → Serial → smplserl.llb → LabVIEW <->Serial.vi を使う方法があります(**図3-15** 参照)．このviを使ってコマンドを送り応答を見ます．

このviは，使う前にシリアルポートの初期化を必要とするので，この**図3-15** の画面が現れたら，**図3-16** の Serial port init.vi を通信条件に合わせてから一度実行して使います．

● シリアルポートの初期化を考える

実験が終わったら，いよいよプログラム構造について考えます．まず，最初に考えなければならないことは，シリアルポートの初期化です．

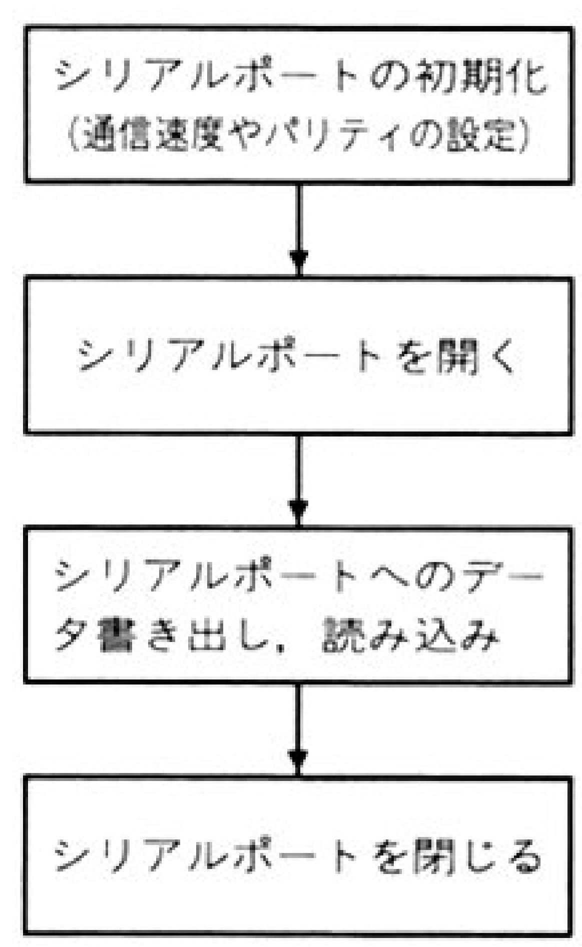
これから作る計測機器に合わせて，通信速度やパリティの設定，フロー制御をどのように設定するかなどを，計測器専用viとして作っておけば便利です．

RS-232C など，シリアルポートを使ったプログラムでは，おおよそ次のフローにそったプログラムである必要があります(**図3-17**)．

今回作った初期化専用viには，**図3-18** に示したようなものを作りました．

通常，シリアルポートの初期化に関するviは，接続

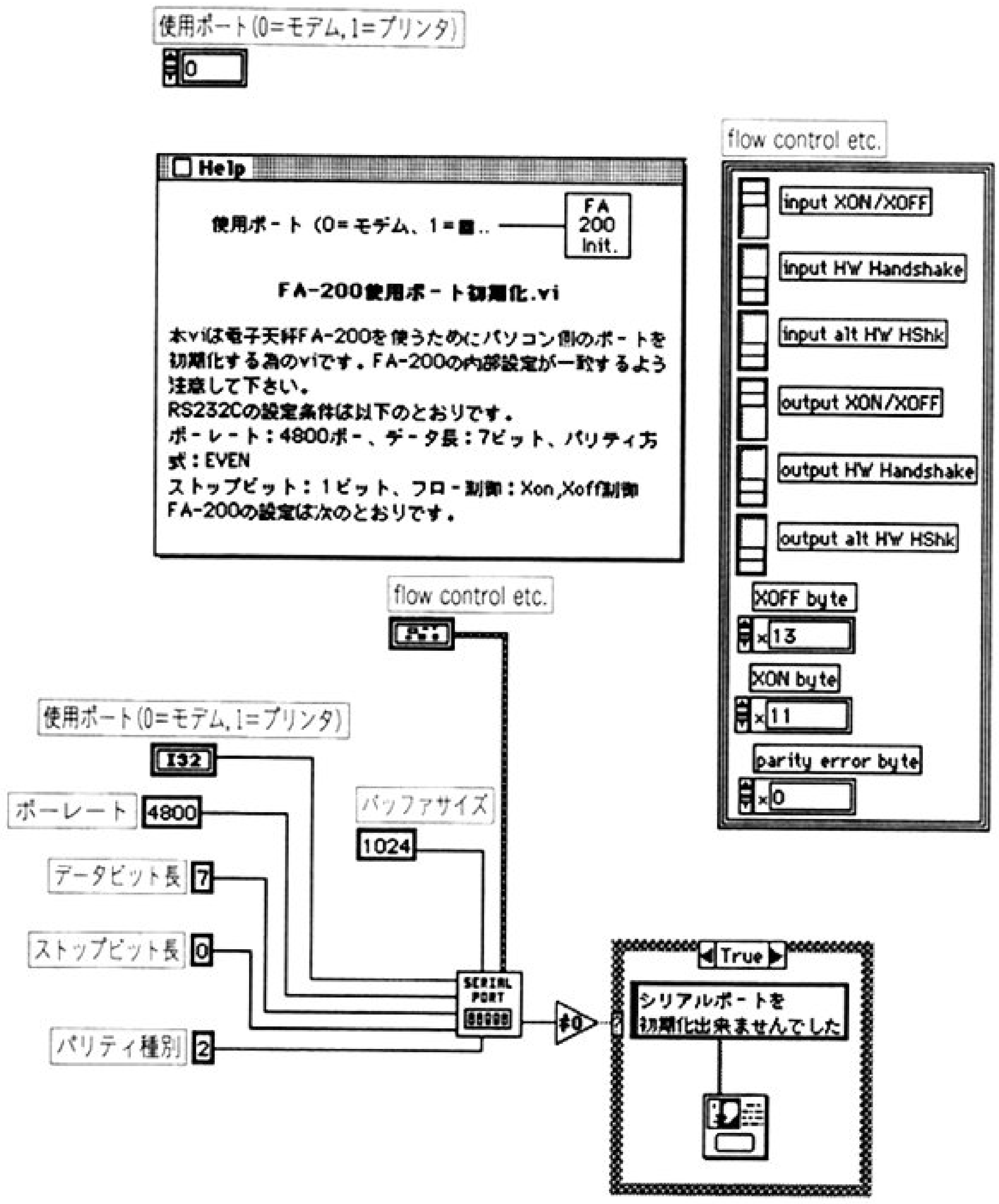
〈図3-17〉
フロー制御の手順



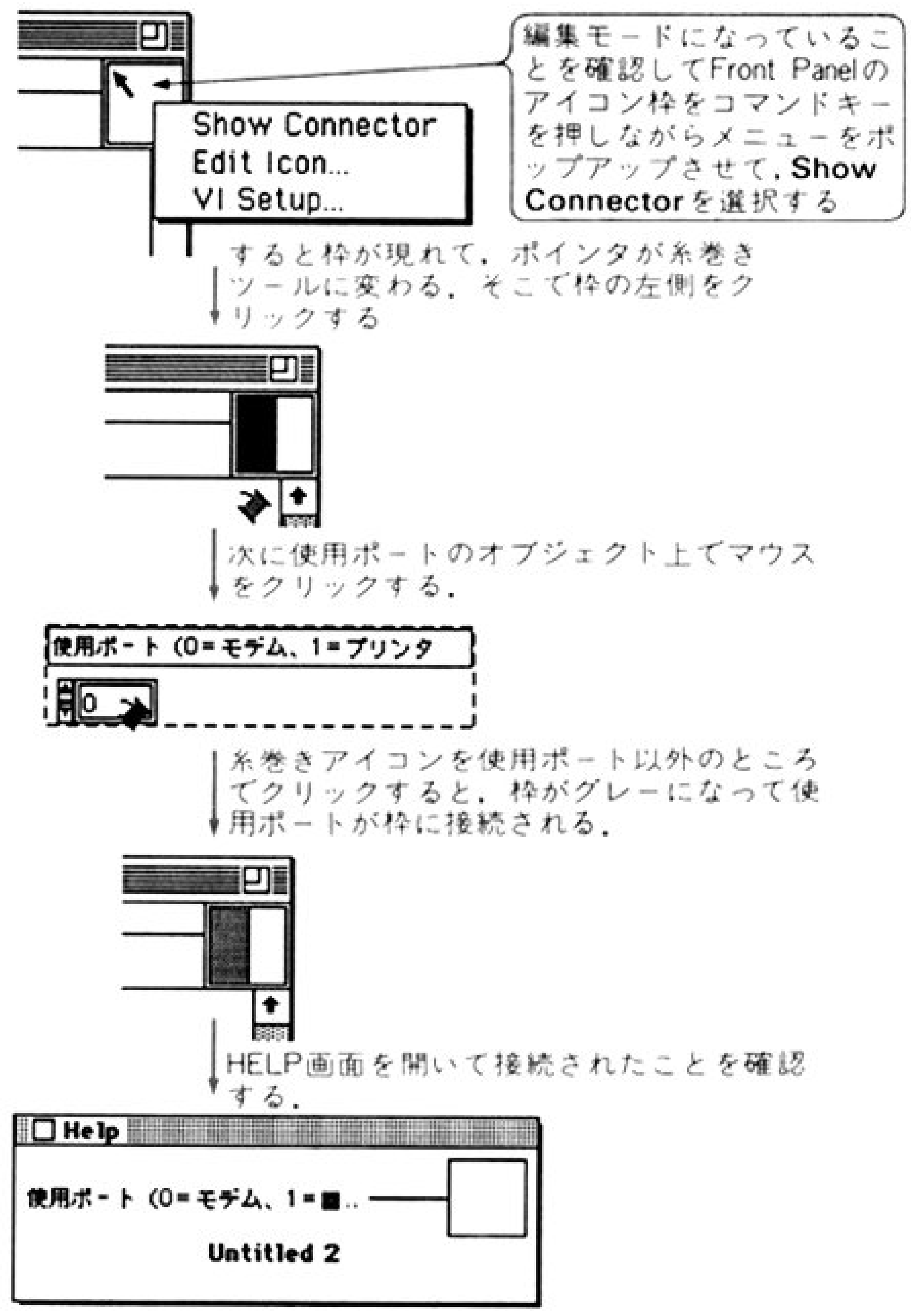
ポートが上位のviから与えられればよいので，この接続ポートのみ外部に接続できるようにフロントパネルの右上のアイコンに接続します．接続方法は，**図3-19** の手順にて接続します(すみません，**図3-19** はまだアイコンの図柄を登録していません)．

また，今回はフロー制御は XON, XOFF と呼ばれるソフトフロー制御を用いていますが，マウスで設定を

〈図3-18〉 初期化専用の vi



〈図3-19〉 シリアルポートの接続手順



行った後はOperateメニューからMake Current values defaultを選択して一度viを保存してください(この方法で，制御器の現状の設定状態を記録して初期値とするわけです)．

この動作を忘れると，再度このviを開いたときに，フロントパネルの設定が設定前に戻ってしまいます．

アイコン枠へのアイコン図案の登録は，**図3-20**の方法で登録します．このとき，三つの枠すべてに図案を登録しておいた方が無難です．

モニター表示可能な色の数が変わったとき，アイコン図案を登録していないと，上位viのアイコンで代替表示され，実にややこしくなってしまいます．無理に作図しなくても，とりあえず文字だけでもOKです．

● コマンドを送るviを作る

さて，次にコマンドを送るviを作ってみます．**図3-21**の手順で，オブジェクトを選んでアイテムの追加を行い4つのコマンドに対応したメニューを選択可能なポップアップを作ります．

また，**図3-14**を参考にして，その他のオブジェクトも配置してラベルを付けてください．

さて，フロントパネルができ上がったら，今度はダイヤグラムを作ります．ダイヤグラムは，**図3-22**を参考に作ってください．

ローカル変数は，シーケンス構造で異なるフレームへ変数を受け渡す接続点のようなもので，今回は使用ポートとコマンドを次のフレームへ渡しています．

使用ポートは，コマンドデータを書き込んだ後に読み取るフレームで使いたいために受け渡しました．

また，コマンドは送ったコマンドデータに対応した受信データから値へ変換するケース分け(条件分け)を行いたいために受け渡しています．

図3-22でやっていることは，送信コマンドに対応したコマンド文字列(ASCIIコード)にターミネータであるCR(改行)とLF(ラインフィード)をくっつけてシリアルポートへ書き出す動作を表します．

● 受信データを解釈するviを作る

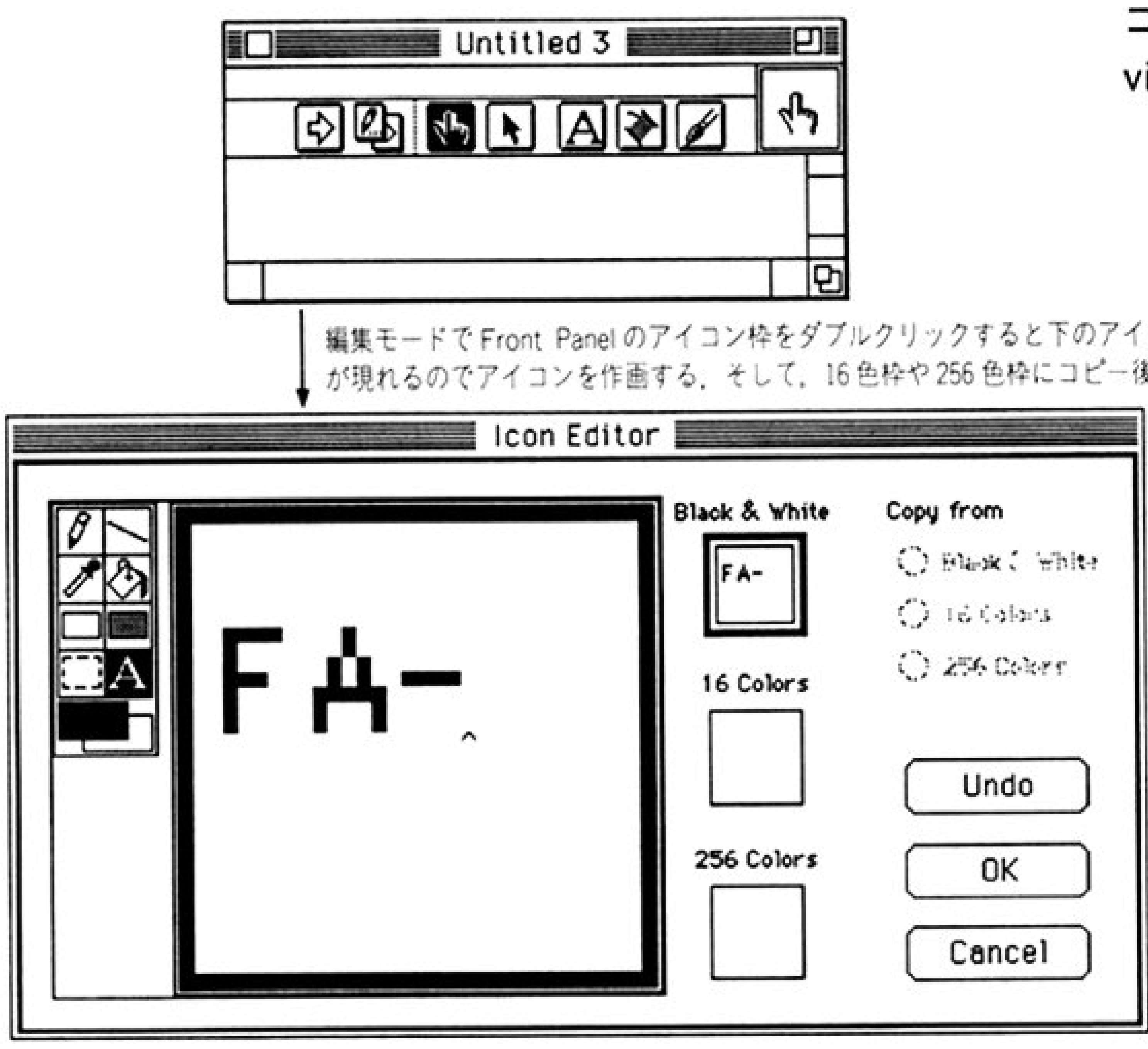
さて，これでコマンドは送りましたので，今度は電子天秤がデータを送り返してくるはずです．そこで，コマンドに対応した文字列を次のフレームで受信して必要な要素に分離します．

図3-22のフレーム1の部分を，今までのテクニックを駆使して作るとダイヤグラムは完成します．

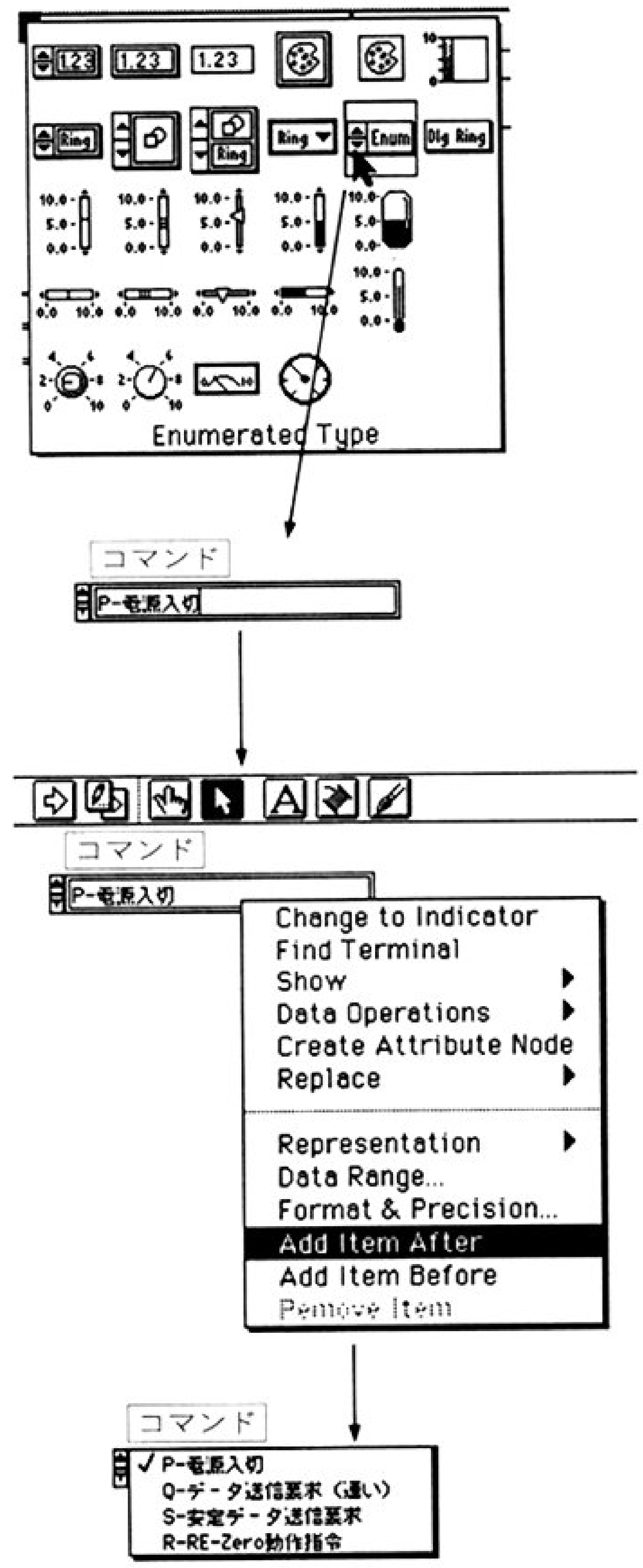
戦い終わって…

さて，一部まだ説明不足のviがありますが，とりあえず，ここまでで電子天秤にコマンドを送ってデータを受信する機能のドライバviを作るところまでこぎ

〈図3-20〉 アイコンの登録



〈図3-21〉
コマンドを送る
vi を作る



つけました(ぜ〜っ，ぜ〜っ)．

次回は，受信データ変換 vi の詳細と，このドライバ vi を使って決まった時間間隔でデータを収集する vi を作ってみたいと思います．

◆ 参考文献 ◆

(1) 多田眞作；グラフィック言語による計測プログラミング，トラ技コンピュータ，1993年8月号．

(2) 田辺敏也；LabVIEW を用いた計測機器とパソコンの接続の実際，インターフェース，1994年12月号．

(3) LabVIEW マニュアル一式，日本ナショナルインスツルメンツ(株)．

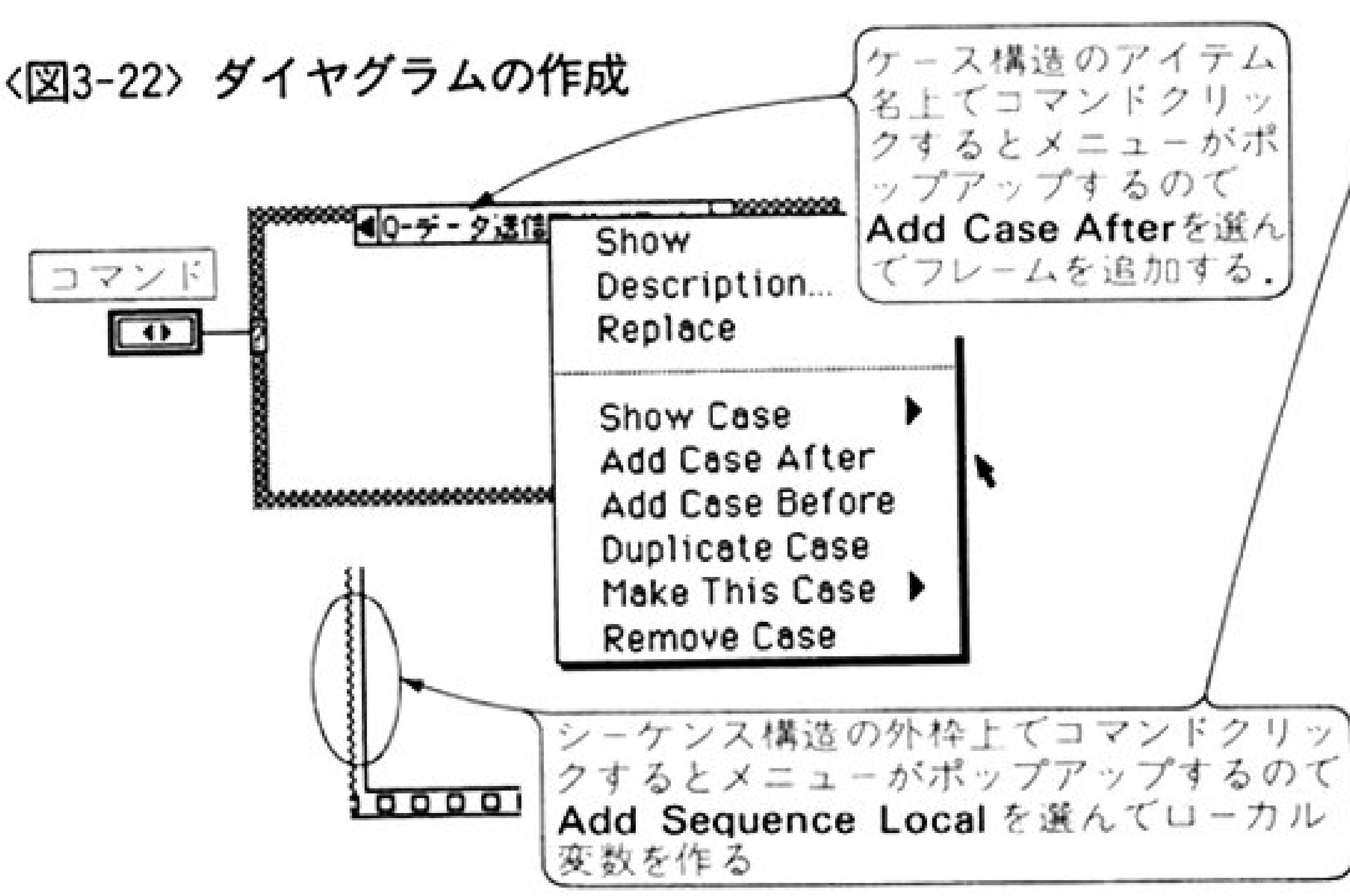
(4) SEAM 92' CD-ROM, MacSciTech, 1992, 国内では LaboFinder (☎ 0425-92-7767) より入手可能．

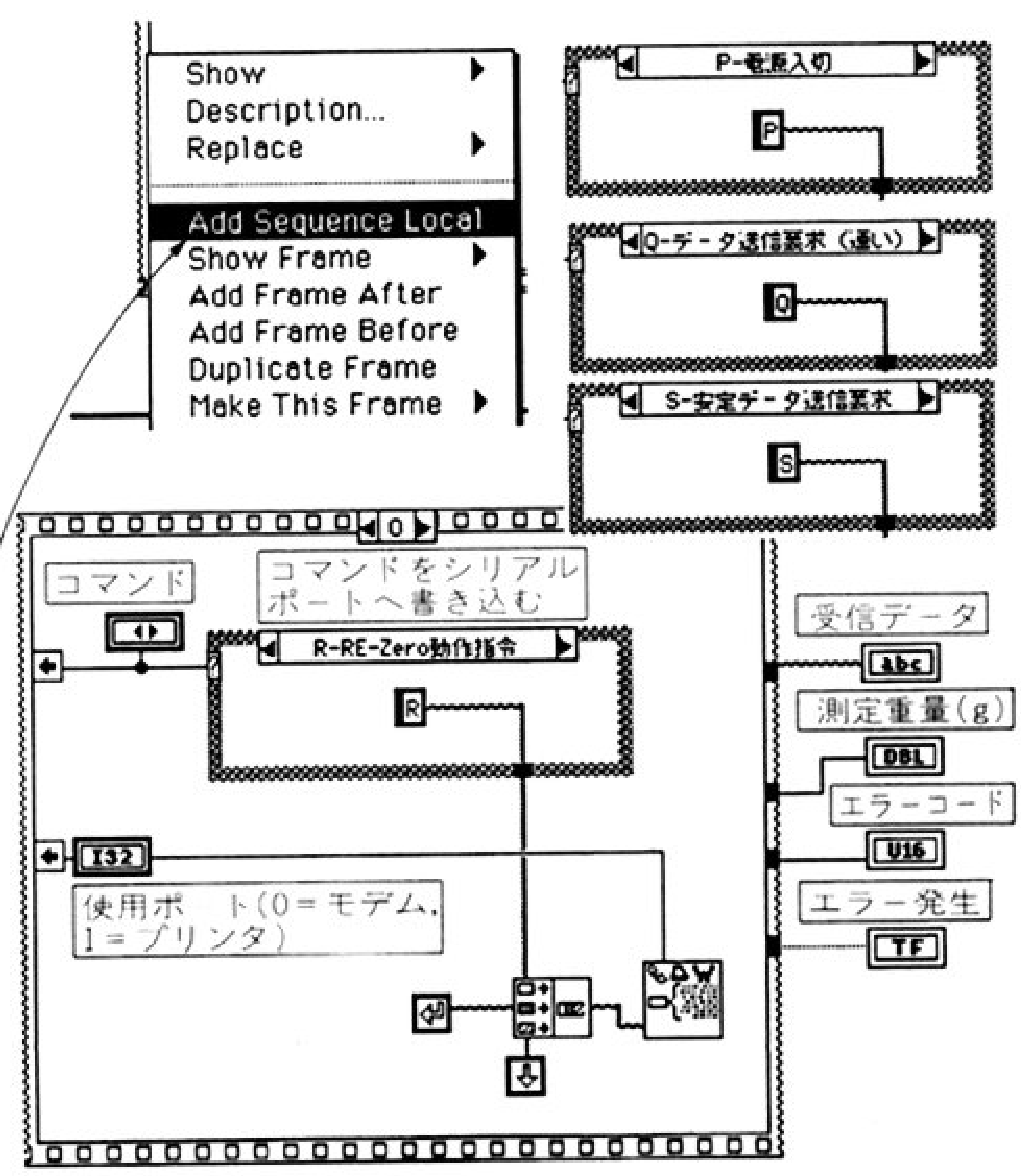
(5) 井上泰典；実践 LabVIEW 3 入門，ソニーテクトロニクス(株)システム営業部．

(6) G. W. Jonson；LabVIEW Graphical Programming Techiniques，McGraw-Hill，ISBN 0-07-032692-4．

(7) ElectronicsUpdate，1994年5月号．

〈図3-22〉 ダイヤグラムの作成





DOS/V パソコンのプリンタポートの怪

COPY コマンドでテキストを PRN に送れない

寺田真一

何かと話題の多い DOS/V パソコンですが，最近ちょっとしたトラブルに遭遇しました．

「DOS/V 機に挑戦するかぎり，多少のリスクは覚悟すべきだ」としかられそうですが，それほど難しいことをしようとしたわけではありません．

ほんとうに簡単な **COPY** コマンドが使えなかったのです．

私の経験が読者のみなさんのお役に立てればと思い，レポートします．特にハードウェア系のエンジニアの方で，PC98 からの乗り換えを計画されている方にとっては有益な情報だと信じます．

● 筆者のパソコンの利用環境

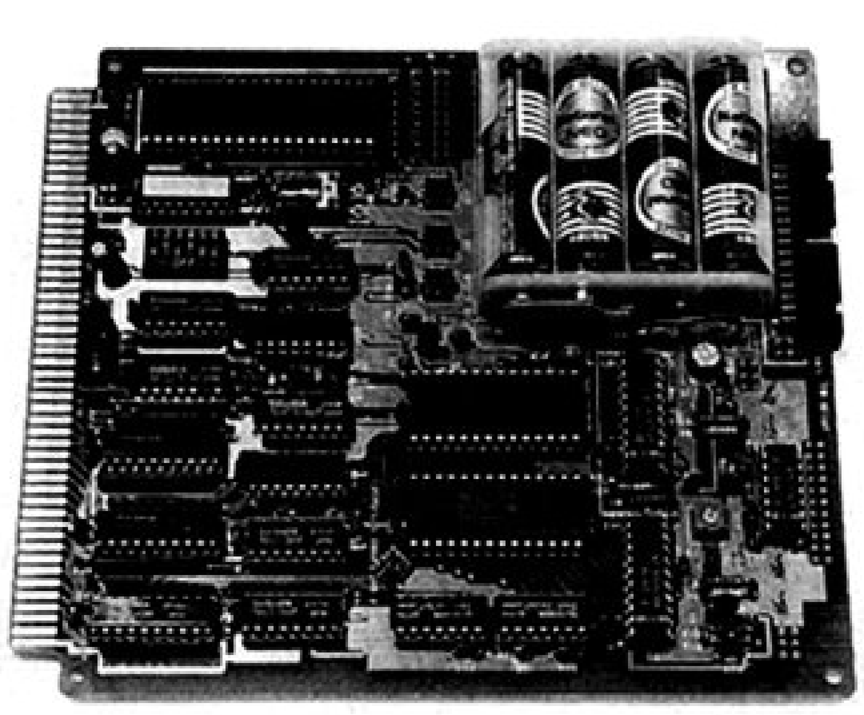
筆者自身は長年 PC98 も IBM-PC も利用してきました．IBM 機では PC-DOS を使うことが多く，ウィンドウズを使う必要性を特に強く感じていないため，DOS/V に変更したのはごく最近のことです．

C 言語やアセンブラなど言語系の仕事をするだけなら IBM-PC のほうが有利です．というのは，IBM 系には優秀なフリーソフトウェアが多く，できるだけお金をかけずにソフトを揃えることが可能です．

といっても，そのフリーウェアの存在するデータベースが海外にある場合が多いため，総合的な通信コストを考えると，フリーウェアを大量に収録した CD-ROM を購入するほうが得策かもしれません．

● PC98 を使ってきた理由

にもかかわらず，PC98 も利用してきた理由は二つあります．



〈写真 1〉PC98 用拡張ボードの例

第一は日本語処理であり，第二は周辺機器や拡張ボードの問題です．

PC98 が事実上の標準機であったつい先日までは，何といっても日本の多くの周辺機器メーカや，オプションボードのメーカが，PC98 に目標を絞っていました．そのため，PIO ボード，A-D/D-A ボード，リレーボードを揃えて何か FA システムを構築するには，PC98 を使用するよりほかに方法がなかったからです．

さあ，ここで DOS/V の登場です．漢字 ROM を持たない IBM-PC でありながら漢字処理を可能にした OS である DOS/V は，コンパックの低価格作戦，IBM のリストラ，Windows 3.1 のリリース，486CPU の低価格化，円高など，多くの要因がプラスに作用し，本来の目的以上の効果を発揮しました．

そして PC98 一色だった日本のパソコン地図を大きく塗り替えはじめたのです．

周辺機器メーカやオプションボード・メーカが，これをだまって見ているはずがありません．IBM-PC 対応ボードを続々発売するに至ったことは，今更説明するまでもないでしょう．

DOS/V パソコン，つまり AT 互換機のシェアが高まるにつれ，日本では今まで入手が困難であった AT バス用の特殊な I/O も続々輸入されています．

今後は IBM-PC を使った各種制御システムも容易に準備できそうです．

日本語とオプションボードの問題が解決すれば，もはや PC98 にこだわる必要はほとんどありません．ここで「まったく」ではなく「ほとんど」と表現した理由は人によって異なるでしょう．

ある人は以前からの取引先との関係，ある人は過去のハード/ソフト資産，ある人は英語力の壁，ある人は日本語 Quick BASIC へのこだわり，など様々な理由から完全に脱 PC98 とはならないのです．

ある官庁に勤務する友人の話ですが，古くなった PC98 の更新用に IBM-PC を予算申請をしたところ，「今までのソフトをむだにするのか！」と大蔵省から大目玉をくらったそうです．

DOS/VのCOPYコマンドがおかしい

筆者の場合，前述の理由はちょっと変わっています．それはPC98はオプションボードの取り外し装着が簡単であるということです．IBM-PCの場合，ボードの差し替えはカバーを開ける必要があるため，例えば**写真1**のようなPC98用のスタンドアロン・データロガー兼用A-D変換ボードは取り扱いがめんどうです．

諸々の事情でPC98と完全に袂を分かつことができない筆者ですが，Windowsを入れる関係上，長年親しんだIBM機のOSを昨年PC-DOSからDOS/Vに変更したのです．

DOS/Vを英語モードにすることでPC-DOSとして問題なく使用できると考えていた筆者に，大きな落とし穴が待ち構えていようとは想像もつきませんでした．

OSをDOS/Vに乗せ換えて2ヵ月くらい後のことです．

日本では有名なA社製のROMライタをセントロニクスインターフェース経由でDOS/V機に接続し，インテル・ヘキサファイルをROMライタに転送したのです．

COPY A.HEX PRNというのが使ったコマンドで，今まで，つまりPC-DOSの時代には何の不都合もなく使ってきたコマンドなのですが，どんなにガンバッても転送できないのです．

試しにROMライタを外しプリンタを接続しますと，正常に**A.HEX**のハードコピーが取れます．念のため，PC98と当のライタをセントロニクスへ接続してテストしましたが，正常に転送されます．

困りはてて，ROMライタのメーカA社へ電話したのですが，「PC98との組み合わせで正常に動作するなら，DOS/Vパソコンの故障でしょう」と言います．

また，DOS/Vパソコンのメーカへ相談すると，「プリンタで正しくハードコピーが取れるのなら，ROMライタがおかしい」という話なのです．

この時点では，OSたるDOS/V自体に問題があるとは思っていなかったのですが，ROMライタを使うのが，DOS/Vに変更して以来初めてであることに気がつき，念のため，DOS/Vを使わずにPC-DOS 3.0のシステムディスクでIBM-PCを立ち上げてみたのです．

これで復旧するとは思わなかったのですが，まさかの一発OK．

犯人は何とDOS/Vだったのです．DOS/Vは少なくとも**COPY**コマンドに関する限り従来DOSからの上位互換性が保たれていません．

しかしIBMともあろう会社が，そんなプアーなOSの設計をするはずはないと思い，担当部門に問い合わせてみました．

いろいろな経緯はあるようですが，IBMの回答は概ね次のような内容でした．

「DOS/Vは従来のDOSに比べてBIOSの大幅変更があり，**COPY**コマンドに関しては，プリンタポートをプリンタ以外の機器に接続して使用するという部分の考慮が欠落したためか，例えばペーパーエンプティのような信号のチェックを強化したために，プリンタ以外の機器には**COPY**コマンドが使えない状況が出ている．

そして，OADGグループという仕様に基づいているため，もはやBIOS変更はできない」というような内容でした．

経緯はともかく，セントロニクス接続ではROMライタが使えません．

残された方法はRS-232Cで接続するように変更することですが，ほかに事情があって筆者はRS-232Cが使用できないため，このA社のROMライタを使う時は，PC-DOS 3.0のシステムディスクで起動しています．古いDOSを保管しておいて助かりました．

DOS/Vパソコンはその互換機の豊富さから，かなり規格が統一されていると思われがちですが，結構不揃いな部分もあります．

この話題のプリンタポートのアドレスにしても，**278h**のマシンもあれば**3BCh**のものもあります．これを知らずに3日間も悩んだこともあります．

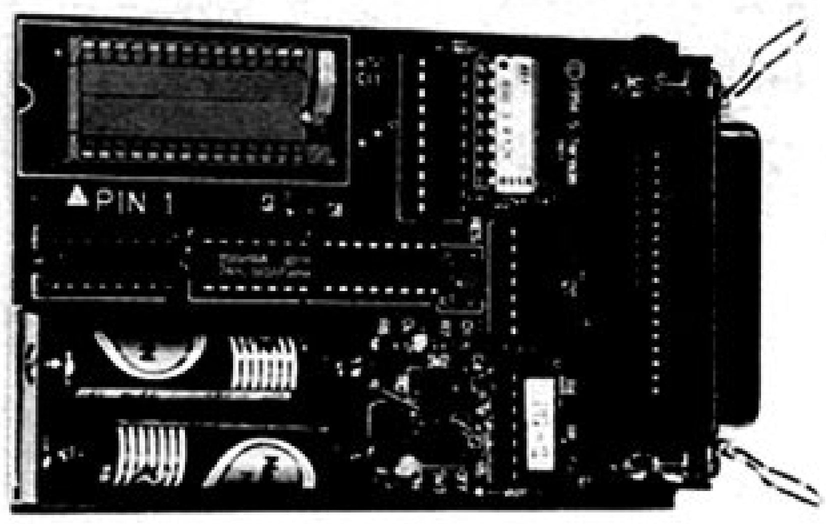
ROMライタに限らず，プリンタポートを利用する

● COPYコマンドでテキストファイルを転送する際の機能の違い

PC98用のMS-DOSやIBMのPC-DOSはコピー先の指定を**PRN**とした場合，BUSYとACK信号のプリンタハンドシェイク仕様に基づき，正常にテキストファイルのハードコピーが取れます．

デバイスとしてROMライタが接続されていれば，このライタの内部メモリにテキストデータが転送されます．

一方，DOS/Vの場合ですが，コピー先の指定を**PRN**とすると用紙切れの信号を大変礼儀正しくチェックしたうえでデータを送るらしく，用紙切れ信号など考慮していないタイプのROMライタには転送することができない，という事態になります．



〈写真 2〉
ROM ライタ

周辺機器はバーコードリーダ，カードリーダ，A-D コンバータなどと，かなりの種類があるようです．特に PC98 と異なり，IBM-PC のプリンタポートはパラレルリードが可能なため，PC98 以上にその種類が多いそうです．

しかし，今回の経験を演繹すると PC-DOS で動作したそれらの機種が DOS/V 上で正しく動作するとは限らないという推定が成り立ちます．

問題となった A 社の ROM ライタも，IBM-PC にも対応すると書かれているのですが，DOS/V では **COPY** コマンドも使えないことがわかりました．A 社は「DOS の上位互換を考慮していない IBM に問題あり」との見解ですが，「DOS/V には対応していません」くらいの注意書きがあってしかるべきだと思うのが，多額の電話代と時間をむだにさせられた筆者の正直な気持ちです．

プリンタポートに接続するハードウェアを利用する機会の多いエンジニアの方は，私の失敗談をぜひ「こやし」にしてください．

IBM-PC と PC98 で使える ROM ライタ

反対に，IBM-PC と 98 の双方の利用のことを親切に考えられた ROM ライタもあります．紹介しておきましょう．

写真 2 は，MT 商事(☎ 06-252-3838)という会社から販売されている EEPROM 専用のライタで，最近続々と新規メーカが参入している 28C64 専用のものです．秋葉原では日立の互換素子 58C65 のほうが入手が容易です．

〈表 1〉 EEPROM ライタ(EE-604 型)の仕様

適用 EEPROM	28C64，58C65(8 K バイト) 日立，NEC，沖，SGS，ザイコー，エプソン各社
適用パソコンと OS	PC98，DOS/V，PC-DOS の DOS モード DOS 2.11 以上．8086～DX4(100 MHz)．プリンタケーブルで接続，各ノート型 PC も使用可
付属ソフトウェア	READ/WRITE/VERIFY/ERASE その他，HEX-BIN 変換ユーティリティ付属．必要メモリ：約 120 K バイト(書き込みの照合は，DOS の **FC** または **COMP** コマンドを併用する)
ファイル形式	バイナリファイル(配置アドレスは自由指定)
動作速度	書き込み：約 12 秒/1 K バイト．読み出し：約 5 秒/1 K バイト(読み取り時間は CPU により 50 %程度変動する)
電　源	単 4 型乾電池 2 本(マンガンで 100 時間以上)
形　状	約 75 mm×110 mm，約 90 g
その他	書き込み時，前データは自動上書きイレース．UV-EPROM 27C64 読み取りのみ可能．27C256 は変換アダプタにより読み取り可．

主に 8 ビット系のワンボードマイコンの ROM 作成に使うもので，アセンブラやマイコンとまとめて教材セットとして販売されています．

たびたびプログラムの修正が必要な学習段階，あるいはデバッグ段階では，イレースすることなくプログラムの上書きができる EEPROM は，確かに UV-EPROM よりはるかに便利です．

筆者は Z80 などの開発にこのライタを使用していますが，おかげで最近紫外線イレーサがヒマをもてあましています．ROM エミュレータにも出番がありません．

このライタには IBM 用(PC-DOS または DOS/V)と PC98 用の 2 種類のコントロールソフトが添付されており，もちろん IBM 用では先の 2 種類のプリンタアドレスも自動選択してくれます．

BIOS 変更の関係で DOS/V における **COPY** コマンドと PC98 系の **COPY** コマンドに大きな差が生じてしまった現在，この EEPROM ライタのように 2 種類のコントロールソフトを添付して販売してくれるような親切なメーカが増えてくれるとありがたいのですが…．

こつこつネットワーキング

⑥ メールと WWW サーバーの設定

編集部

● **Microsoft MAIL とは**

WindowsNT によるネットワークが立ち上がったので，複数のユーザーが使えるアプリケーションとして，Microsoft MAIL をインストールしました．

▶ 10 ユーザー(DOS/Windows 用)…スターターパック……98,000 円

▶ 1 ユーザー(Macintosh 用)(Mac のクライアントを使う場合)……48,000 円

最初にこの二種類が必要です．

WindowsNT Server の中にも最初から Microsoft MAIL が入っているので，クライアント(ユーザー)用だけを購入すればよいのかと思いましたが，そうではありませんでした．WindowsNT に入っているのはすべて Peer to Peer の NT 同士でしか使えないメールソフトで，メールサーバーにはなりません．

WindowsNT Server という名称から，どういう場面でもサーバーになることができると思うのはまちがいのようです．Microsoft MAIL は，LAN マネージャの時代から使われていたもので，セキュリティ関係も独自の方法をとっており，NTFS のファイルシステムともうまく適合しません．

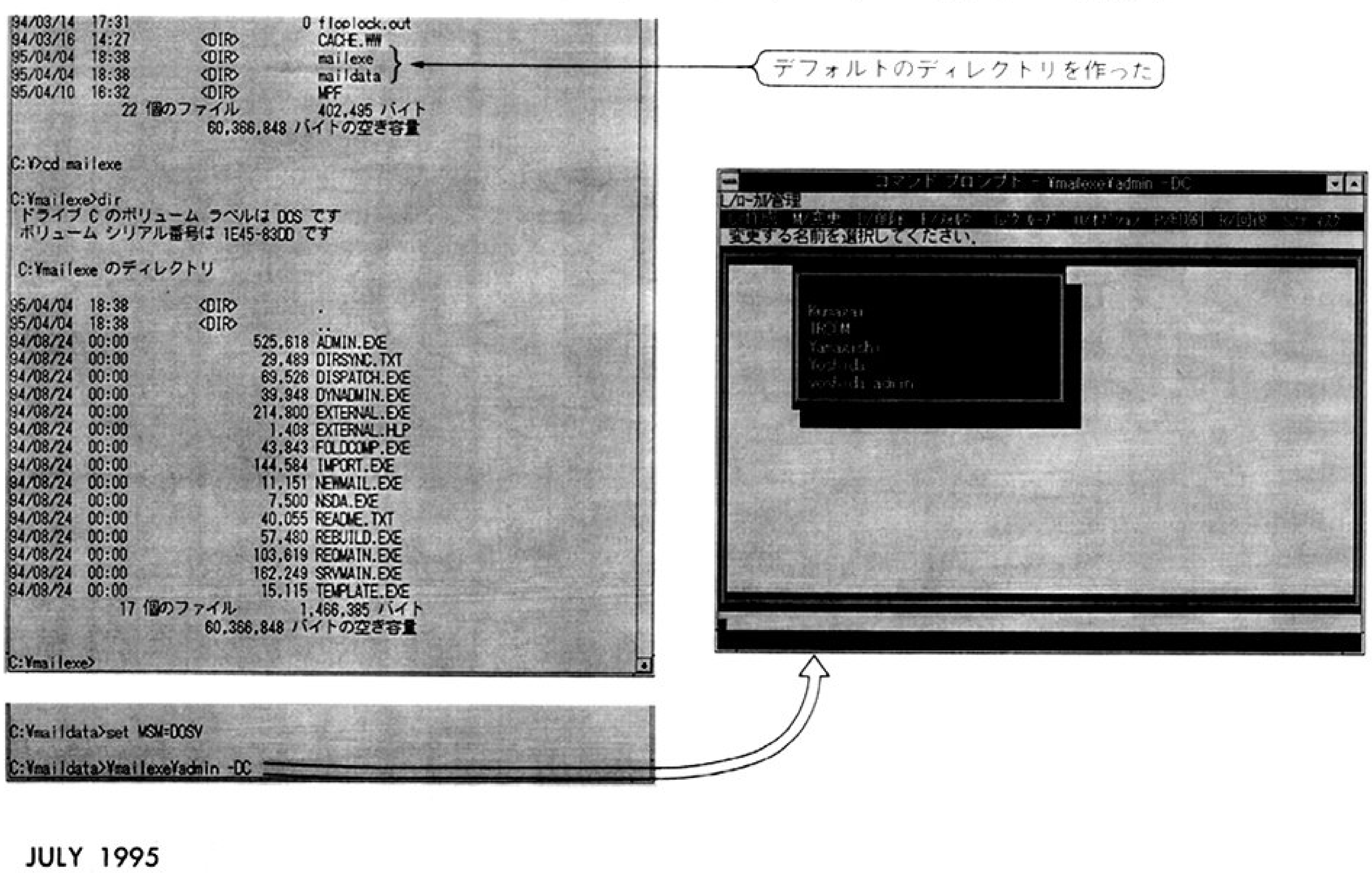
● **Microsoft MAIL のインストール**

まず最初に，クライアントのソフトを WindowsNT Server にインストールします．各クライアントは，M：ドライブをデフォルトのメール用ドライブとして利用します．

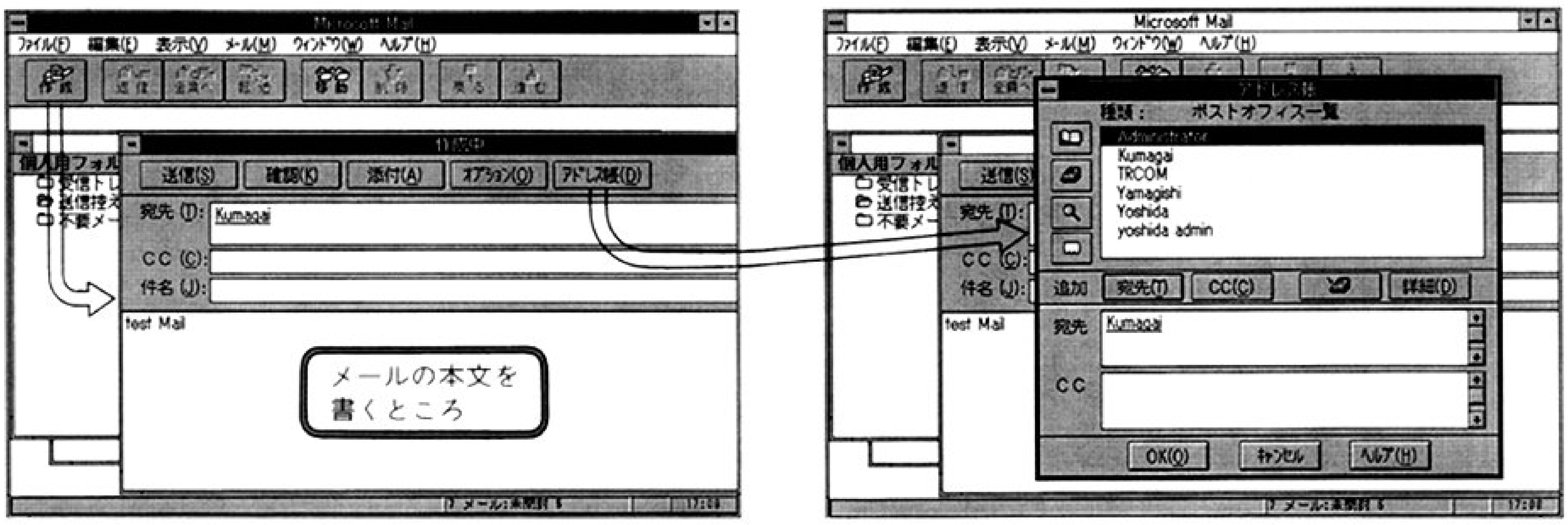
次に，サーバー用ソフトをインストールします．NTFS のドライブにインストールしましたが，動作しませんでした．NTFS では，ディレクトリやファイルの所有権，アクセス権を管理者やユーザーの一人一人について個々に設定できます．これらの設定がうまく合えば，NTFS のファイルシステムでも利用ができると思われますが，**README** などには，何も情報がありませんでした．

UNIX ではグループ **news**，オーナー **news** というように所有権などの設定ができ，セキュリティなども

〈図6-1〉 WindowsNT の FAT ドライブ(C：)にインストールした admin を動かす



〈図6-2〉クライアント側のWindows用Microsoft MAILを使っているところ



しっかり確保されます。

そこで，FATのドライブであるC：ドライブにメール関連のすべてのファイルを転送し，**admin**を動作させました(**図6-1**)。

クライアント側のインストールで，一件だけうまくいかなかったマシンがあります．このマシンでは，ウイルスチェック用ハードウェアが拡張スロットに差さっていました．何度やってもインストールの最後の段階でマシンがハングアップするのです．

結局，このハードウェアを抜くことでインストールはうまくいきました．インストール自体の作業はクリックしていくだけですから簡単です．

● Microsoft MAILの運用

WindowsNT Serverのコマンドウィンドウの中で**admin**を動かして，ユーザー(クライアント)のメールボックスを作ります．つまり，サーバー側でクライアントすべてのメールボックスを集中管理します．

今回は入手しなかったのですが，Macintosh用のメールも，一つのMacintoshにメールボックスを管理するサーバーを作ります．そして，Macintoshのメールサーバーと，WindowsNT Serverのメールサーバーが互いに通信して，メールを交換するという手順になるようです．

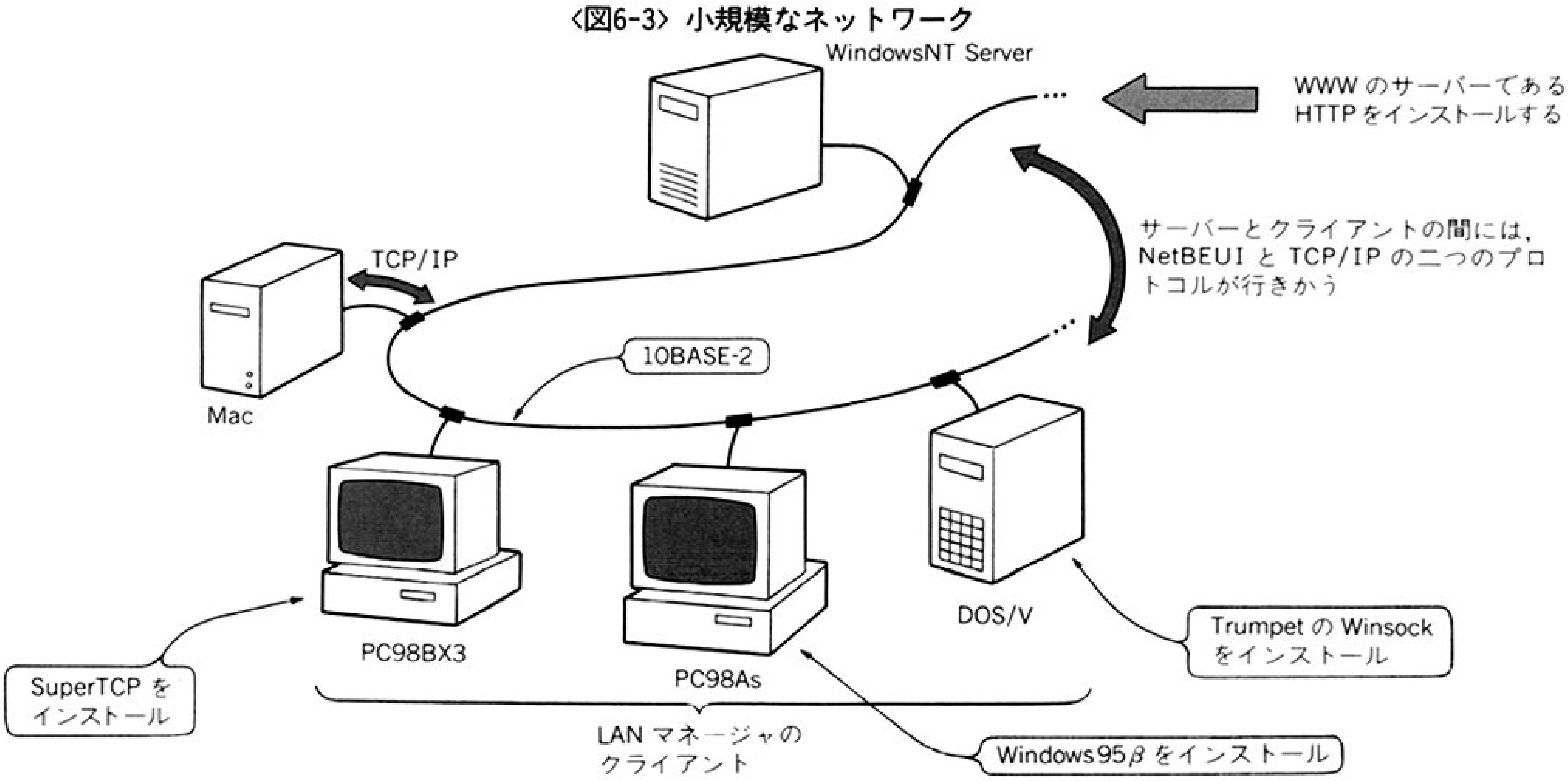
クライアントのメールボックスを作ったら，Windowsのクライアント側では，**図6-2**に示すように，アドレス帳に宛先が表示されます．ここでメールを送る相手を選び，本文に何かを入れて「送信」ボタンをクリックすることで，メールを送ることができます．

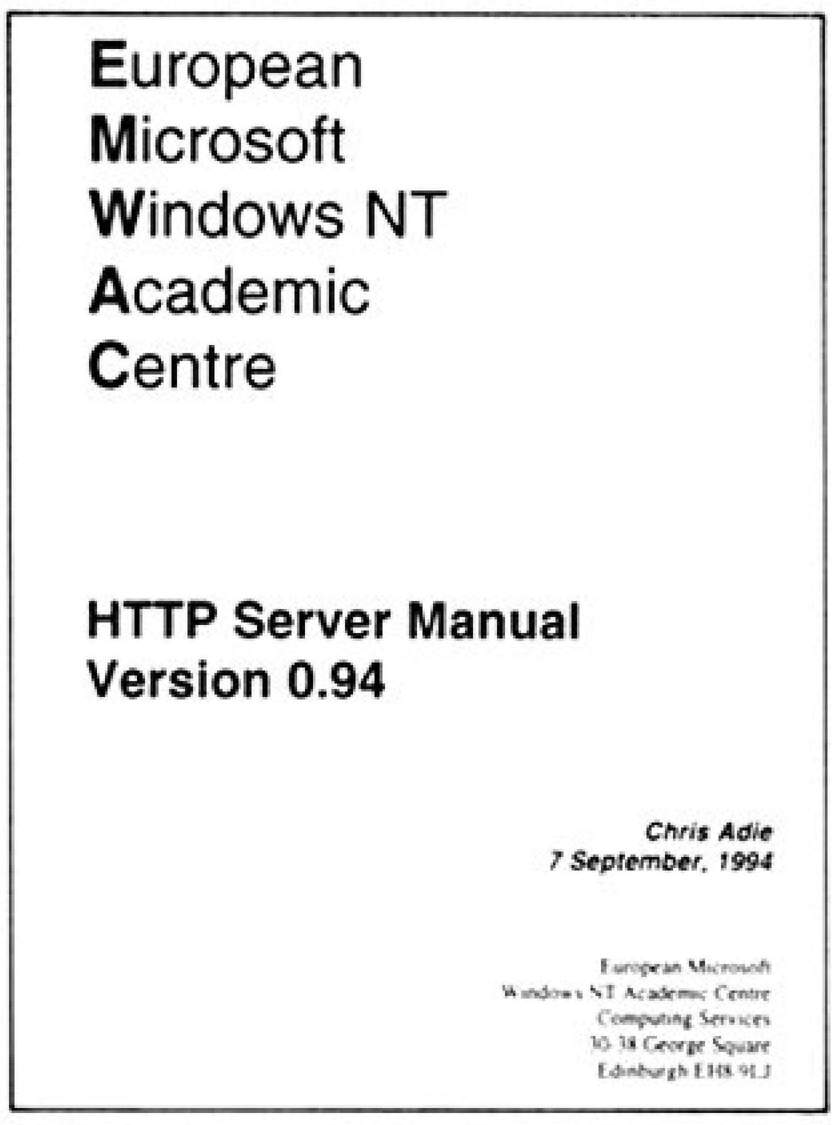
● しかし不便

ファイルマネージャから，**XXX.TXT**というファイルをメールの本文エリアにドラッグ&ドロップすることで，ファイルを貼り付けて，メールを送ることができます．しかし，拡張子のないファイルを送ると受け取った側ではそれがテキストであっても読むことができません(その方法をまだ見つけることができていない)．

さらに，ディレクトリごとメールに貼り付けようとすると拒否されてしまいます．つまり，何でも送れるメールではないようです．

〈図6-3〉小規模なネットワーク




European
Microsoft
Windows NT
Academic
Centre

HTTP Server Manual
Version 0.94

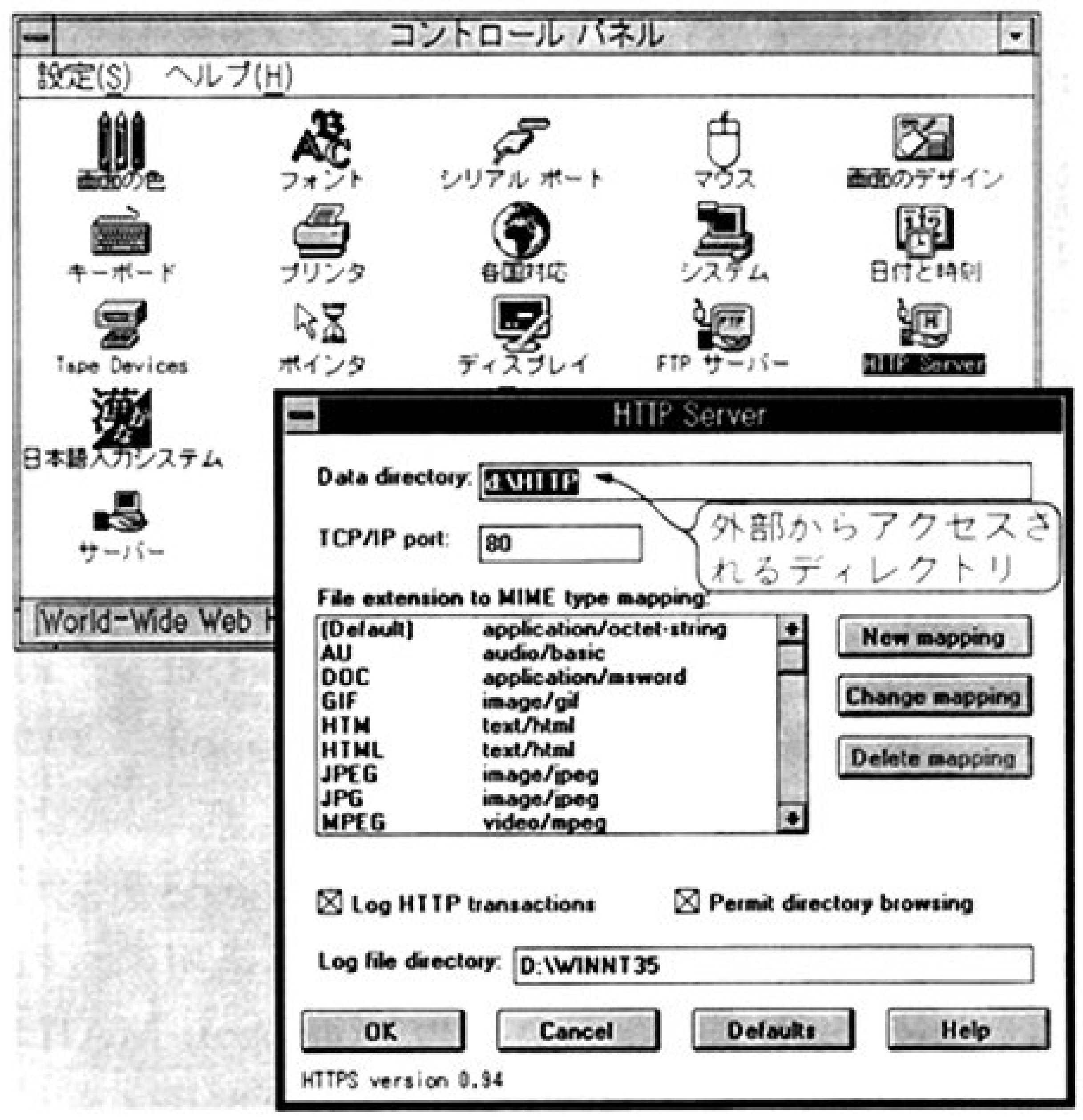
Chris Adie
7 September, 1994

〈図6-4〉HTTPのマニュアル

〈表6-1〉HTTPに含まれているファイル

ファイル名	内　容
HTTPS.EXE	HTTPのサーバープログラム
HTTPS.CPL	コントロールパネルのアプレット
HTTPS.HLP	上記のヘルプファイル
HTTPS.DOC	マニュアル(Word)
HTTPS.PS	マニュアル(ポストスクリプト)
HTTPS.WRI	マニュアル(ライト)
EGSCRIPT.ZIP	CGIのスクリプトサンプル
COPYRITE.TXT	コピーライトの表示
READ.ME	リードミー

〈図6-5〉コントロールパネルにできたHTTPサーバーを動かす



編集部内では，公開ディレクトリにファイルマネージャを使って，必要なテキストファイルやディレクトリをコピーするという方法のほうが簡単なので，Microsoft MAILは使われなくなりました．つまり，メールを送る相手が見えるところにいるという環境では，メールが必要でないという結果になりました．

しかし，もっとうまいメールの使い方はないものかと思い，次回ではCC：Mailをインストールしてみたいと考えています．

● HTTPをインストールする

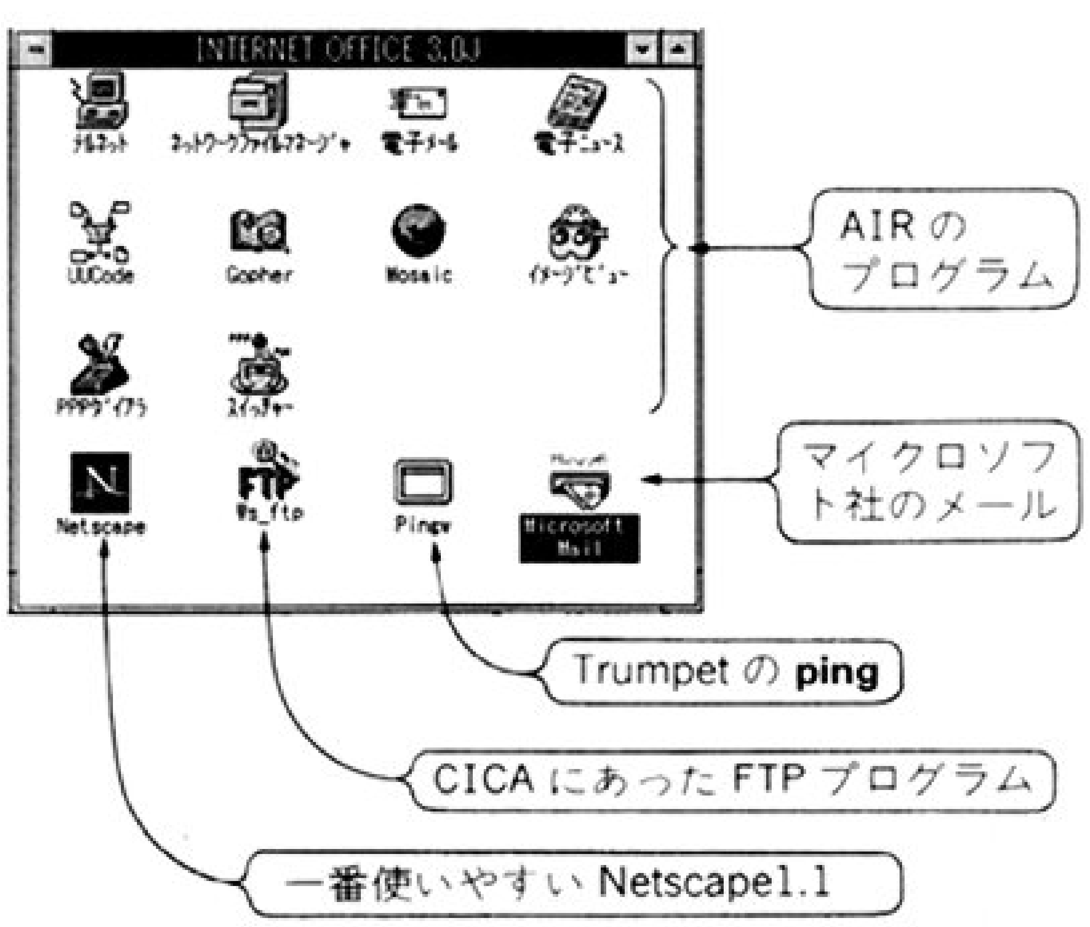
現在，**図6-3**のようなネットワークになっています．LANマネージャではなく，TCP/IPに注目すると，どのクライアントもIPを利用できる環境です．

SuperTCPは，コンテック社が日本語を販売しているVxD(仮想ドライバ)形式のTCP/IPプロトコルをサポートし，その上で動く各種ソフトのセットです．Windows95ではTCP/IPは標準でビルトインされています．

ここで，WWWのサーバーをインストールして，小規模なネットワーク内でもWWWが利用できる実験を行ってみます．

UNIXなどでは，CERN(WWWの発祥の地)のhttpdというWWWサーバーを動かすことが多いのですが，WindowsNTでは，英国のEMWACという団体が作ったHTTPというWWWサーバーが入手できます(**図6-4**)．

このHTTPは，**http://emwac.ed.ac.uk**に最新版がありますが，回線が細いせいかほとんどアクセスで



▼AIRのツールに含まれているツールの一部

テルネット	端末エミュレーション
ネットワークファイルマネージャ(NFM)	ファイル転送とリモートファイル管理
電子メール	TCP/IPインターネットメール
電子ニュース	インターネット・ニュースリーダ
Gopher	インターネット検索ソフト
Mosaic	インターネット，ハイパーテキスト表示ソフト
イメージビュー	グラフィックデータ表示，編集ソフト
PPPダイアラ	モデムの設定，ダイヤルアップ

〈図6-6〉インターネットアクセスするためのツール類

きません．WindowsNT リソースキットなどに古いバージョンが入っていますから，それを使うことにします．Version 0.94 の中には**表6-1** に示すファイルが入っています．マニュアルに従ってインストールすると，コントロールパネルにアイコンが追加されます(**図6-5**)．

● WWW のクライアント

図6-3 に示したように，Windows のクライアントには各社の **winsock.dll** がインストールされています．このレイヤがあることで，様々なインターネットへのアクセスツールが共通に使えるようになります．

- ▶ CICA の CD-ROM の **/winsock** にあるツール(無料)
- ▶ AIR などの市販のインターネットアクセス・ツール(19,800 円)
- ▶ SuperTCP など，**winsock.dll** とインターネットアクセス・ツールがまとまっている市販のソフト(約 40,000 円)

図6-6 は，AIR 製の「INTERNET OFFICE 3.0J」のグループのアイコンです．このソフトの最大の特徴は日本語化されていることです．CICA などには英語版しか上がっていませんから，**ftp.iij.ad.jp** などの FTP サイトから日本語の使えるツールを入手したり，**xxx.ini** ファイルを自分でフォントを変更するなど，とにかく日本語を利用するには手間がかかります．

この図にある「電子メール」は，インターネットで使われる POP というプロトコルを使う汎用のものですので，マイクロソフト社独自の Microsoft MAIL とメールのやりとりはできません．この「電子メール」を使うには，POPer と呼ばれるメールサーバーを呼ぶように設定しなければなりません．

DOS/V のクライアントでは，Trumpet をインストールしましたから，このディレクトリの中にある **Hosts** ファイルに，

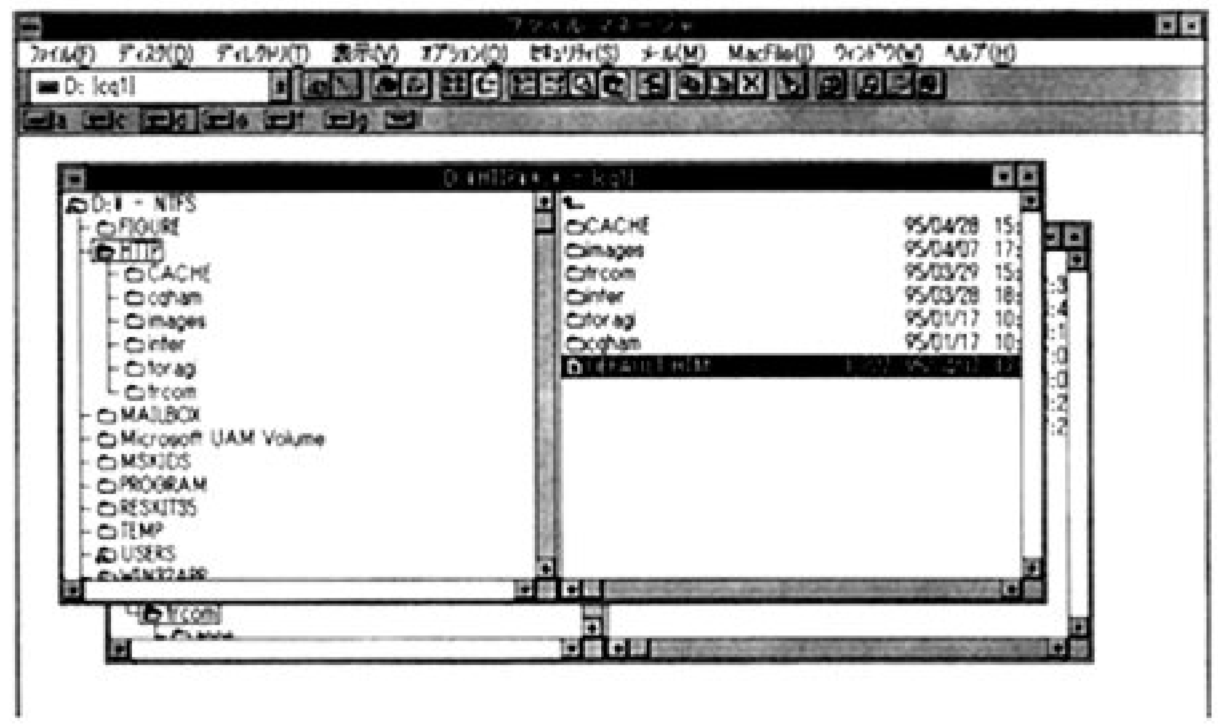
202.17.215.202　trcom　www.trcom

というように，WindowsNT Server に別名をつけておきます．そして，WindowsNT Server の HTTP ディレクトリに，**DEFAULT.HTM** という HTML のファイルを置いておきます(**図6-7**)．Mosaic のサーバーなどで **Welcome.html** と呼ばれるような，アクセスされたときに開く最初の HTML のファイルです．

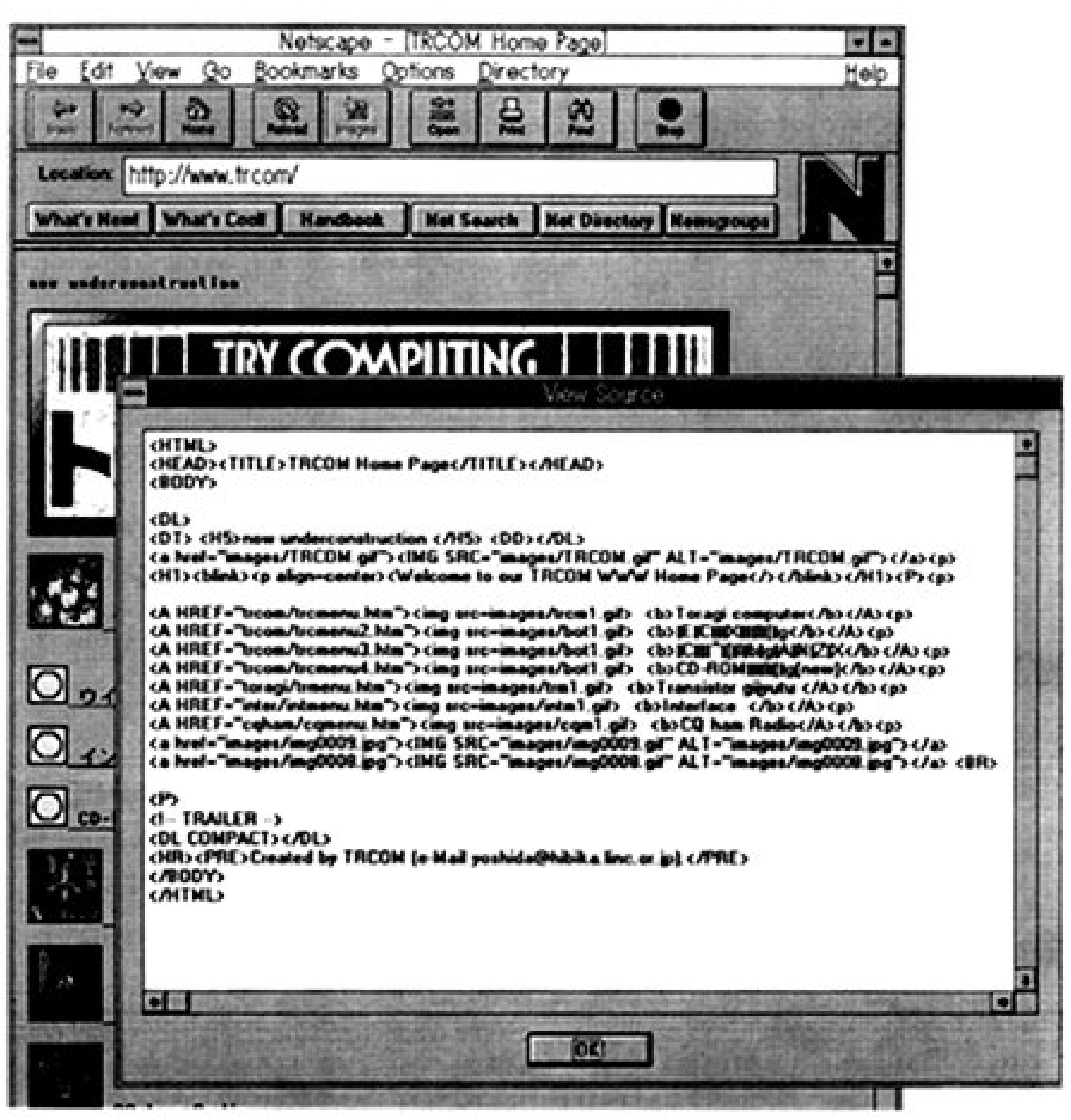
その内容は，**図6-8** に示すように簡単な HTML で記述されたテキストファイルです．アンダーラインの引かれた項目をクリックすると，**図6-9** のようにハイパーリンクされたファイルをたどっていくことができます．

つまり，WWW を運用すれば，社内掲示板などの代わりに使えるのです．もちろん専用線でインターネットにつなぐことができれば，世界中から**図6-8** や**図6-9** のファイルがアクセスできるようになりますが，

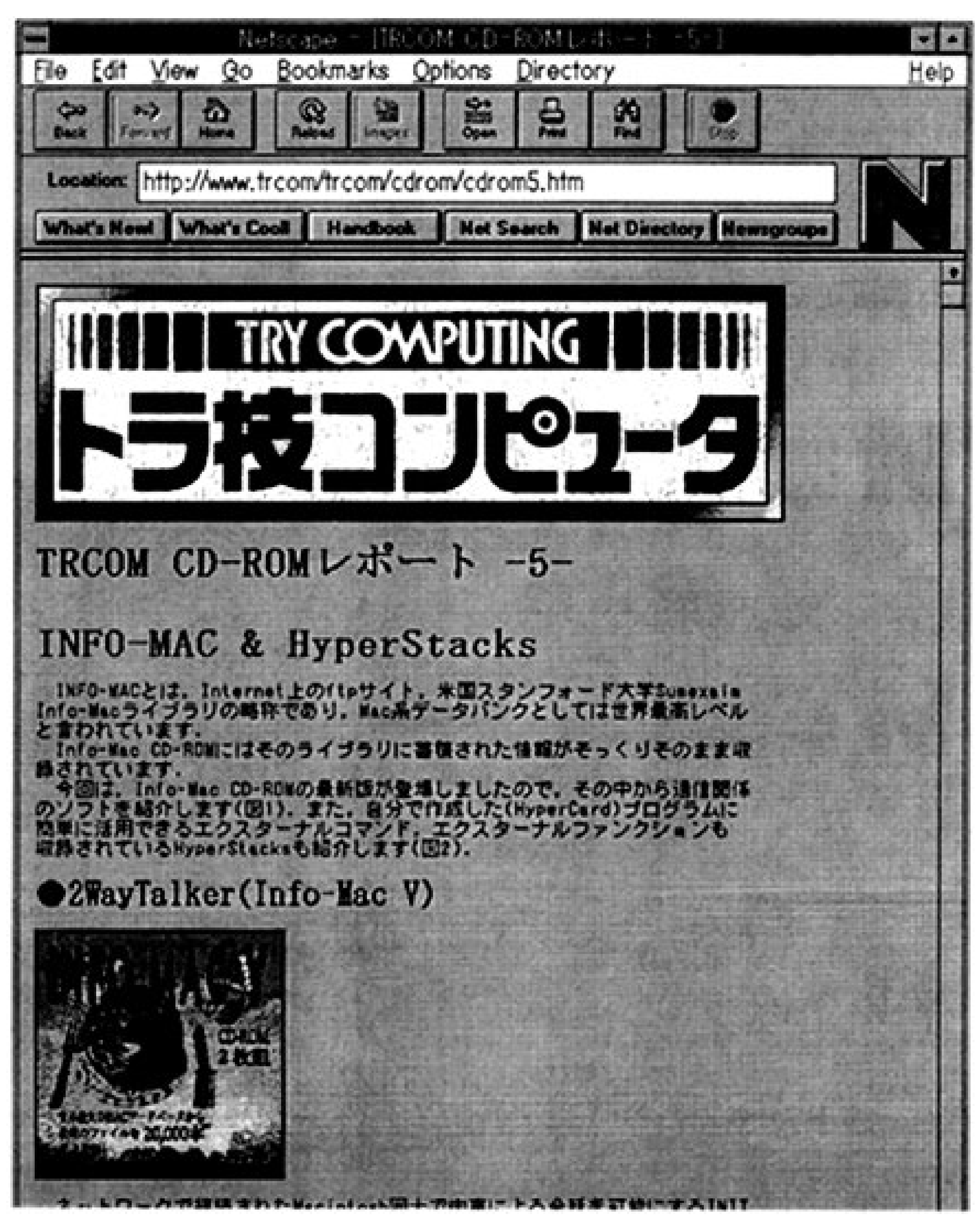
〈図6-7〉 DEFAULT.HTM の位置



〈図6-8〉 DEFAULT.HTM の中身



〈図6-9〉 ハイパーテキストの表示例



REXX言語によるOS/2入門

3 ビジュアルなプログラミングができるVX-REXX

山下直彦

Windowsの普及により，エンドユーザーに優しいプログラムの作成が可能になってきました．しかし，それはプログラマへの負担を極端に増やしています．その軽減策として，Visual BasicやVisual C++，最近ではDelphi(ボーランド社)で代表されるVisualなプログラミング環境が注目されています．

この流れは，MS-Windowsの世界に限定されず，UNIX，OS/2などの世界でも進行しています．今回は，REXXのビジュアル化について触れます．

GUIを利用するために，イベントドリブンなプログラムが強調されていますが，これは必ずしも，GUIの専売特許ではありません．キャラクタベースのCUIプログラムであっても，インタラクティブなプログラム(もっとも身近な例はメニュープログラム)では不可欠の要素です．

実用的なプログラムは，フロントエンドの設計におけるイベントドリブンな部分と，データを実際にアクセスする部分である従来のシーケンシャルな部分を組み合わせて作るようになります．

REXXにおけるVisualプログラミング

前回，**RexxUtil**パッケージの使用に触れました．非常に基本的な要素については，このパッケージに含まれる関数を使用してPM(プレゼンテーションマネージャ)のプログラミングが可能です．つまり，**RxMessageBox**関数や**WorkPlaceShell**関数を利用します．

また，これを補強するものとして，IBMのEWS(注1)の**VREXX2**があります．これには各種Window，Dialog，さらにGraphics関数が含まれています．

これらは，OS/2のREXXの拡張を利用して，簡単なGUIプログラムの制作を簡便に行えるわけですが，設計段階はビジュアルではありません．

設計段階もビジュアルなプログラミングを可能にするものとして，サードパーティの製品がいくつか出ています．その一つは，後ほど扱うWatcom社(注2)のVX-REXX，HockWare社のVisPro/REXX(これについては，次回扱う)やGpf System社のGpfRexxなどがあります．

VX-REXXによる例題(その1)

VX-REXX(注3)を使って，簡単な例題のプログラムを用いながら説明しましょう．

● 準備

このプログラムをディレクトリ，**C:¥VXR1**に収納するものと仮定します．最初に，このディレクリを作成しましょう．

デスクトップのWatcom VX-REXXのアイコンをダブルクリックし，Projectフォルダをダブルクリックします．すると，VX-REXX Projectテンプレートが現れます．

さらに，このテンプレートをダブルクリックすると，**Window1.VRX**，**Window1.VRY**，**Project.VRY**が現れます．

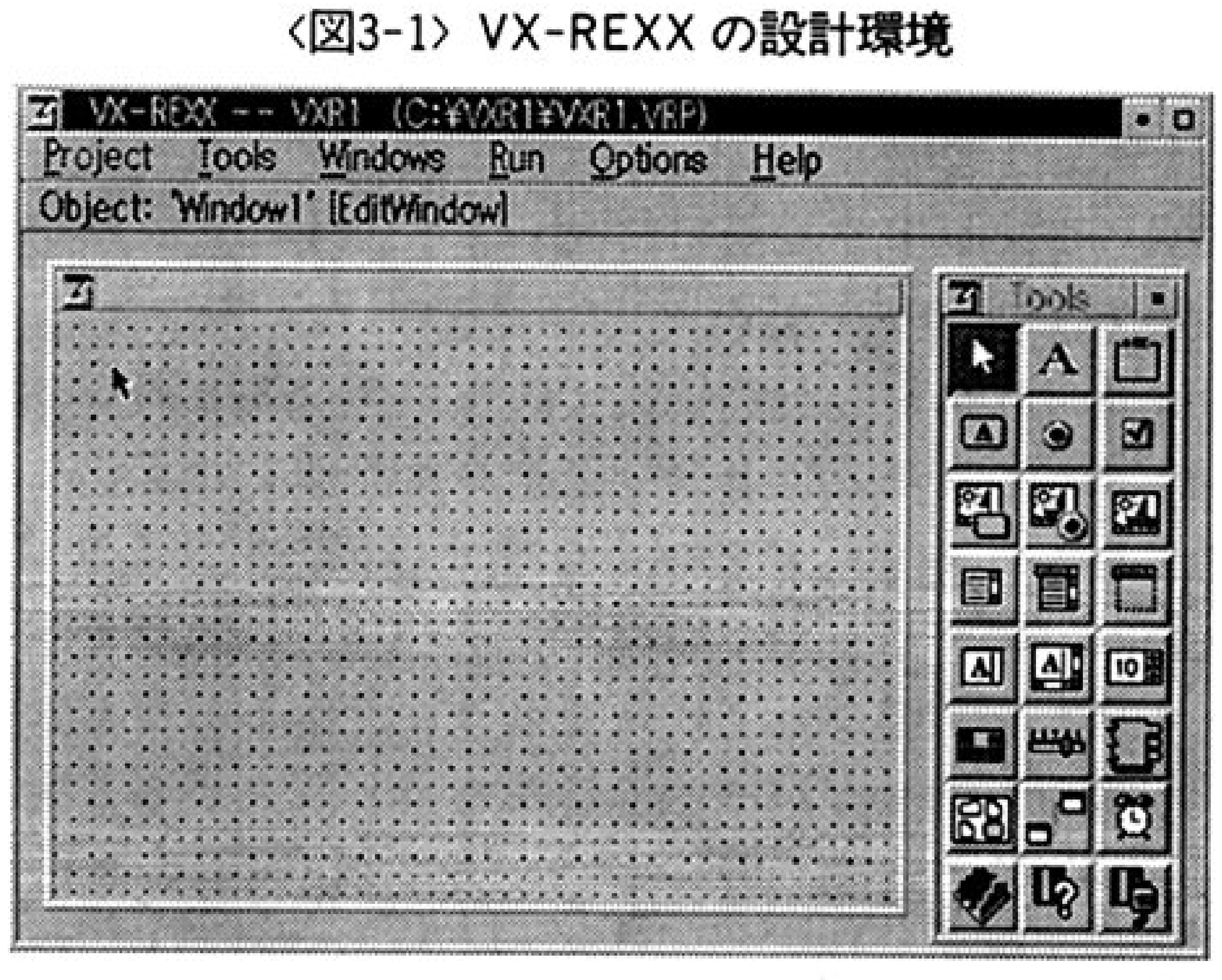
Project.VRPを，先に作成したディレクトリ**C:¥VXR1**にコピーして，名前を**VXR1.VRP**に変えて，これで準備が整います．

● プロトタイプの作成

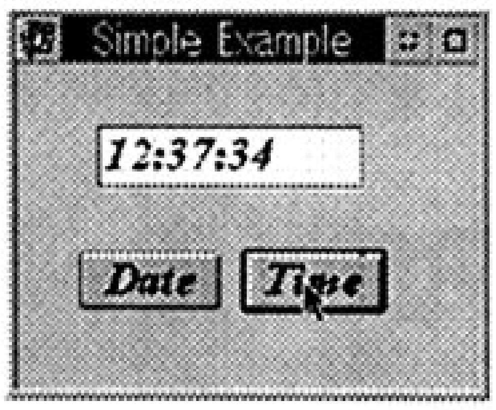
この拡張子**VRP**のファイルをダブルクリックすると，設計環境プログラムが起動されます．

図3-1に示したのは，このVX-REXXの設計環境で

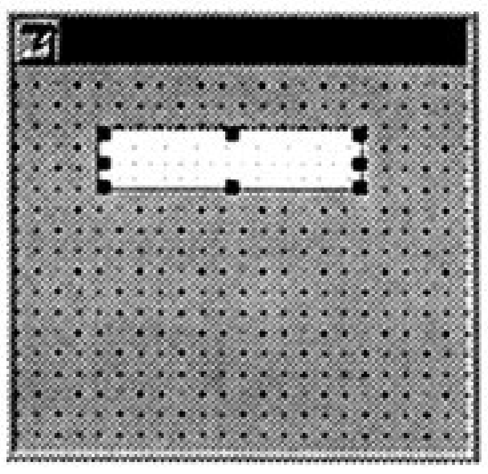
〈図3-1〉 VX-REXXの設計環境



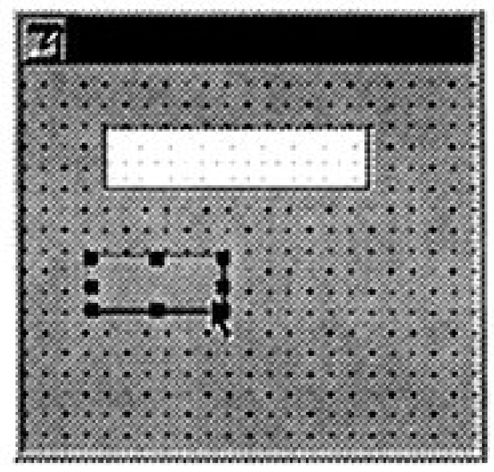
〈図3-2〉 日付けと時刻を表示するプログラム



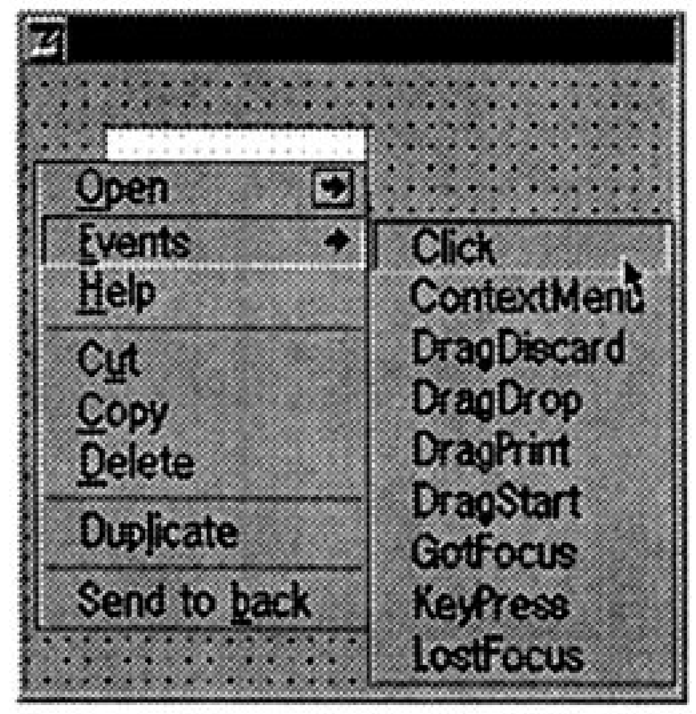
〈図3-3〉 ボタン1を作る



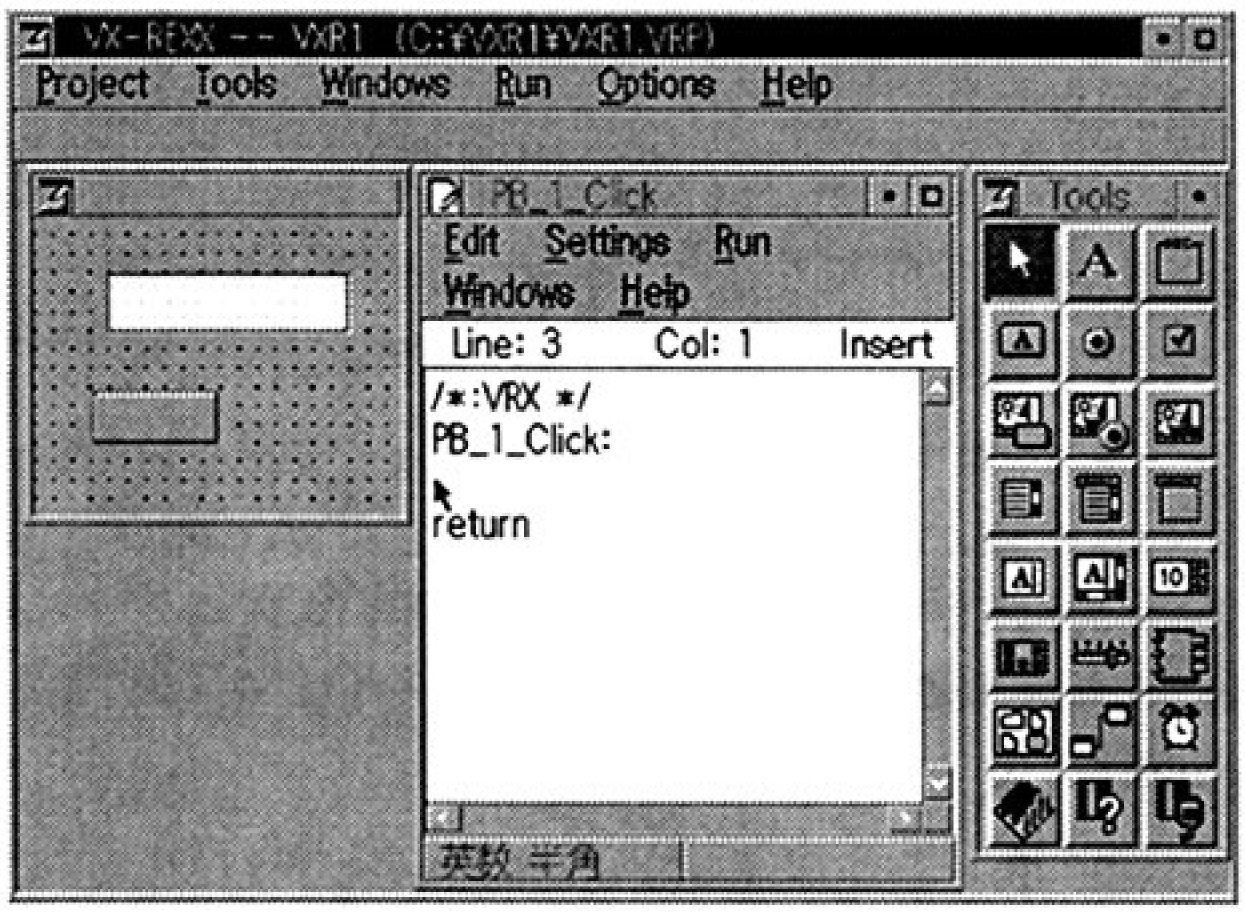
〈図3-4〉 プッシュボタンを作る



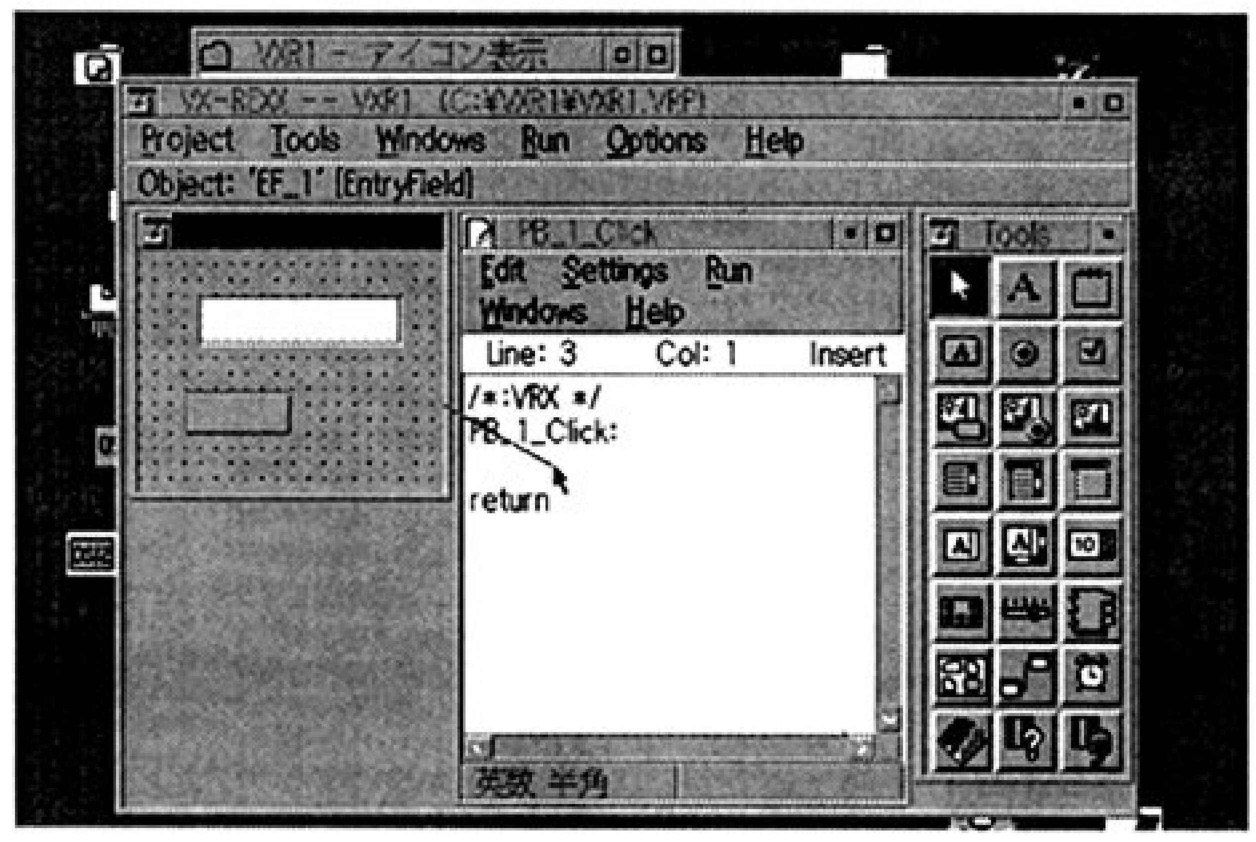
〈図3-5〉 Events のメニュー



〈図3-6〉 プッシュボタンのスクリプトを書く



〈図3-7〉 スケルトンへドラッグしてくる



す．メニューの Options の Show hints の状態(デフォルト)にしておくと，メニューバーの下部に Hint が表示されて間違いが少ないでしょう．

この例題で試みることは，**図3-2** に示すプログラムを作ることです．**Date** と **Time** と名付けられたボタンを押すと，それぞれ日付けあるいは時刻を表示するというシンプルな例です．

既にお気付きでしょうが，わずかに用語法に違いがあるものの，VX-REXX は Visual Basic と非常によく似た点をもっています．Visual Basic の **form1** に対応するのが **Window1** で，通常ドットによる格子が付加されています．

この画面に右側の Tools パレットの Object (VisualBasic のコントロールに対応)を張り付けていきます．この方法も，Visual Basic と同様です．

まず，画面設計として **EntryField** の **Object** をマウスボタン1でクリックして選択し，マウスカーソルを左側のウィンドウ画面に移動します．すると，カーソルが十字に変わります．

張り付けたい左上の位置で再度ボタン1を押し，そのまま斜め右下方にドラッグすると点線で囲まれた長方形が現れます．ボタン1を離せば**図3-3** のように8個の黒い選択マーカが現れ，フィールドの範囲を示します．もし必要ならこれらをドラッグすることで大きさを調整できます．ほかの場所をクリックすると選択マーカは消えます．

なお，デフォルト(GridAlig が選択状態)では，格子点で位置が調整されますので，位置調整をするときに利用しましょう．

フィールドには自動的に名前が付けられます．今の例では "**EF_1**" となるでしょう．

同様にプッシュボタンのオブジェクトを張り付けます(**図3-4**)．これは "**PB_1**" と名付けられます．

画面設計は終わっていませんが，プログラムの要点をつかむ意味で，このボタンを押したときのコードを作成しましょう．

マウスカーソルを "**PB_1**" の領域内に置き(Hint で確認できる)，マウスボタン2を押します．メニュー(**ContextMenu**)が現れます．"**Events**" を選択すると，さらに詳細なメニューが現れます(**図3-5**)．

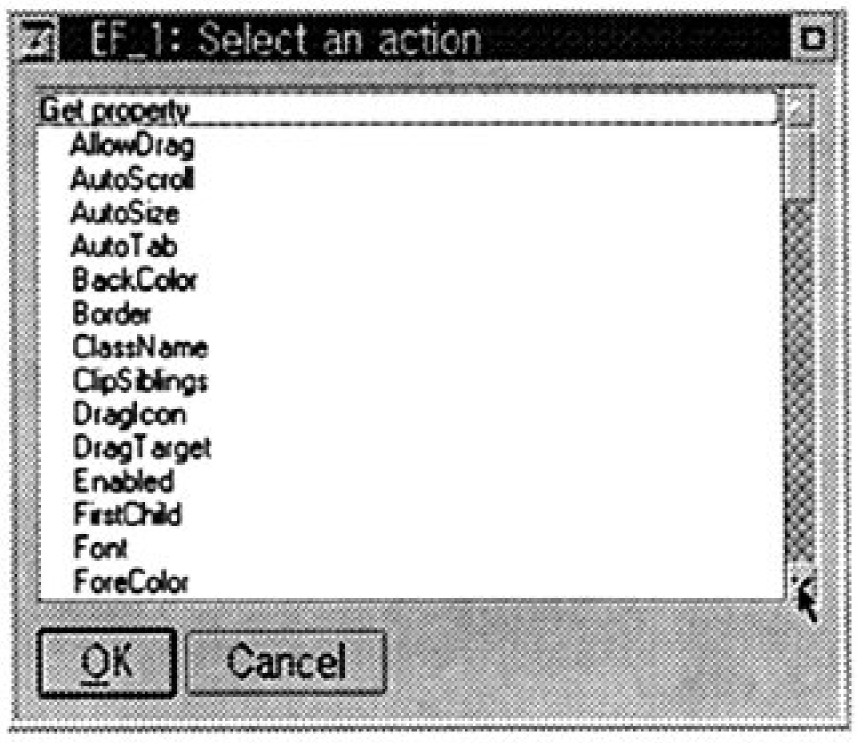
"**Click**" を選択すると，**図3-6** に示すように "**PB_1_Click**" のサブルーチンのスケルトンが現れます．ここに **PB_1** が押されたときの操作を記述します．

Visual Basic と異なる点のひとつはこの先です．これから "**EF_1**" に日付けを書き込むコードを作りたいので，"**EF_1**" をボタン2で選択し，"**PB_1_Click**" のスケルトンにドラッグします(**図3-7**)．

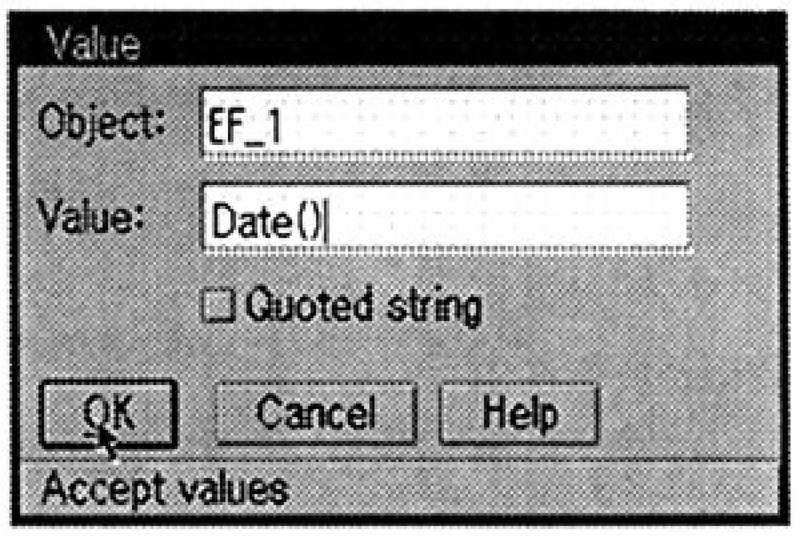
そうすると，**図3-8** に示すような "**EF_1 : Select an action**" のリストボックスが現れます．これは，"**Event info**"(注4)，"**Get property**"，"**Methods**"，"**Set property**" と分類されています．"**Set**" の中から "**Value**" を選択します．

次に，**Value** 設定画面が現れます．**Object : EF_1** は最

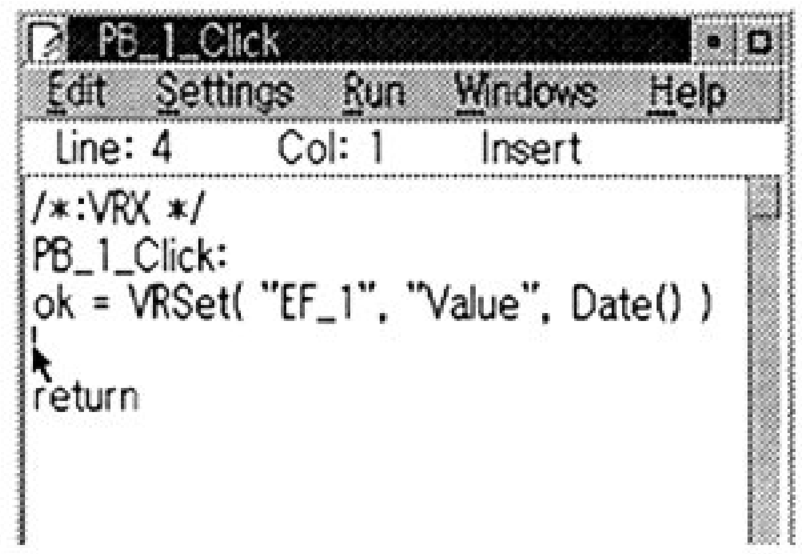
〈図3-8〉 actionのリストボックスの内容



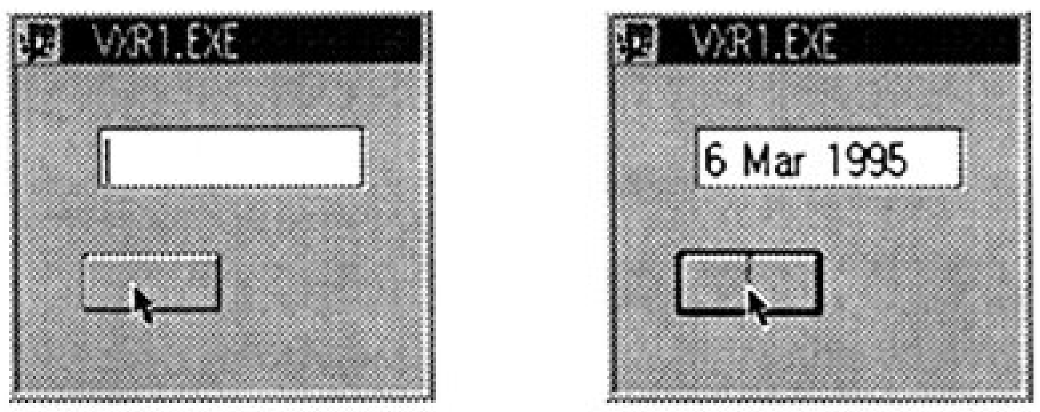
〈図3-9〉 Valueの設定



〈図3-10〉 コードが入った

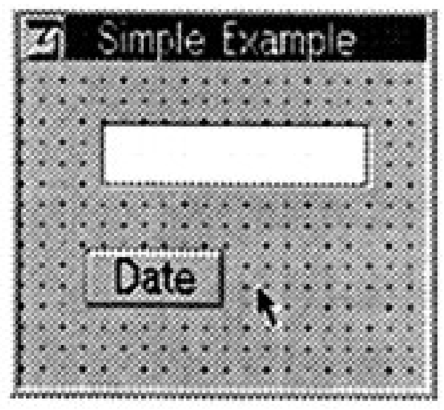


〈図3-11〉 動作を確認

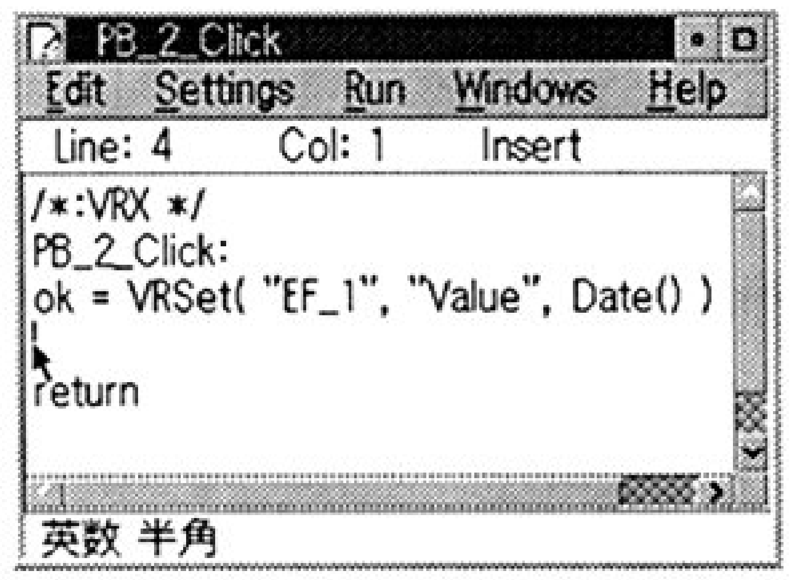


(a) クリックすると　(b) 日付けが表示される

〈図3-12〉
キャプションを記入



〈図3-13〉 PB_2_Clickには
PB_1_Clickのコピーが入る



初から埋まっています．**Value：** のフィールドにREXX関数の**Date( )**を**Quoted string**のチェックボックスをクリックしてこの設定を解除します(**図3-9**)．

“OK” ボタンで設定を終わると，**図3-10**のようにREXXの1行が生成されます(この1行の開始位置は，直前のカーソルの位置)．

この文の意味は，オブジェクト **“EF_1”** の **“Value”** プロパティに**Date( )**の値をセットする〔先に**Quoted string**を設定のままにした場合は文字列として **“Date( )”** がセットされるので注意〕，成功したかどうかは変数**ok**(**1**のとき成功)に代入されます．

以上で最低限のプログラムができましたので試してみましょう．Ctrl+**R**キーを押すと，**図3-11**(a)に示したウィンドウが現れます．**“PB_1”**を押すと，**図3-11**(b)のように日付けが表示されます．

● 仕上げ

試行を終わって，設計環境に戻り，プログラムを仕上げましょう．

ALTキーを押しながらオブジェクトをクリックするとキャプション入力ができます．タイトルバーに**Simple Example**と，**“PB_1”** に**Date**と入力しましょう(**図3-12**)．

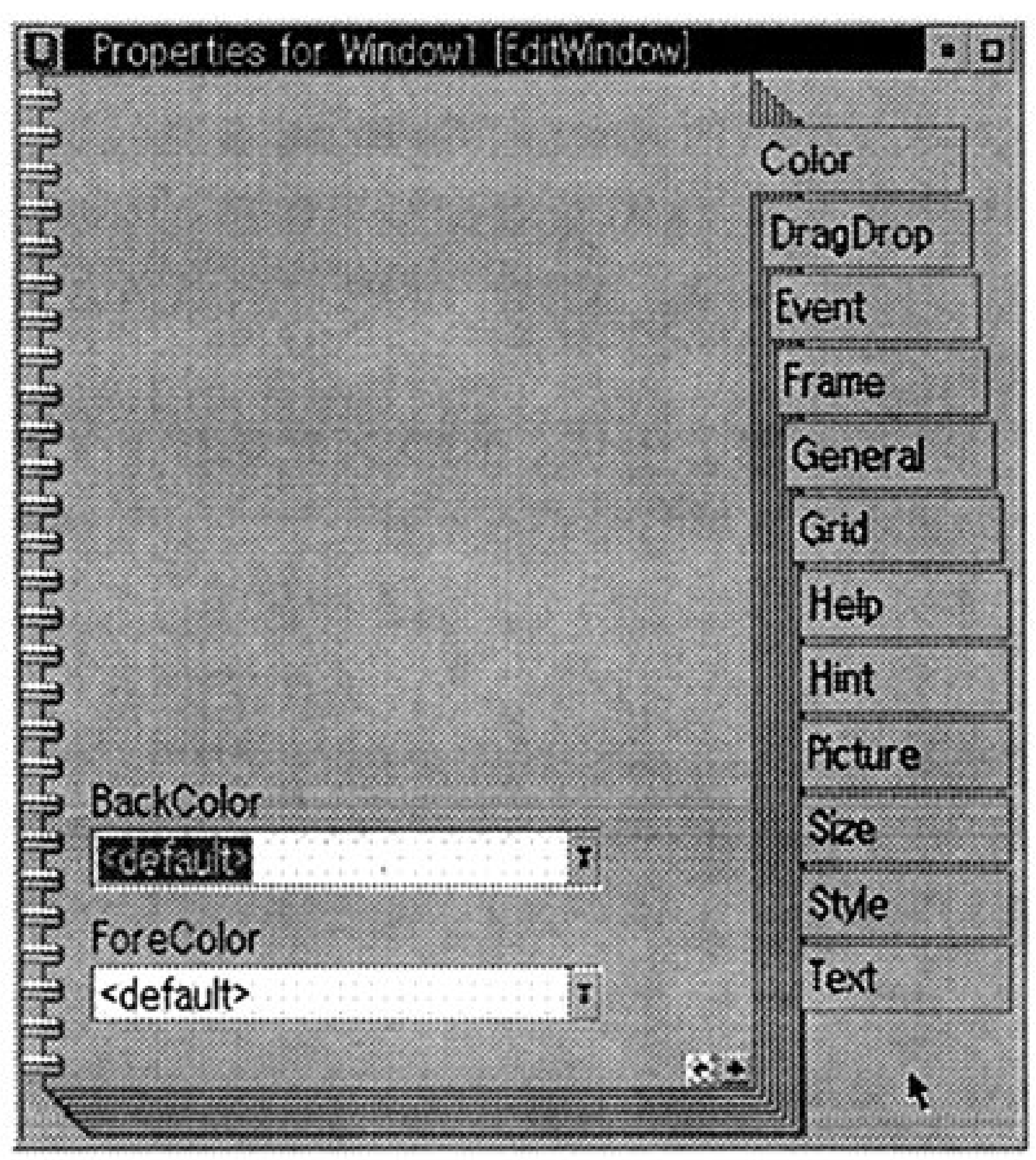
さらに，プッシュボタンをもうひとつ作成します．そのため**“PB_1”**を複写することにします．**“PB_1”**をマウスのボタン2でクリックすると，メニューが現れます．ここで**“Copy”**を選択します．二つ目のPBの左上の角の位置をマウスのボタン2でクリックすると再度メニューが現れます．ここで**Paste**を選択すると**“PB_2”**と名付けられたボタンが生成されます．このキャプションを**Time**に変えます．

“PB_2” を押したときのコードは，**図3-13**のように**“PB_1_Click”** のコピーとして生成されていますので，**Date( )**の部分を**Time( )**と替えることで充分です．

各々のオブジェクトのプロパティの初期設定は，対象のフィールドをボタン1でダブルクリックすると現れるノートブックで行います．**Window1**の設定ノートブックを**図3-14**に示します．これは慣れるまで，やや拡大して見るようにしたほうがよいでしょう．ノートブックの性格上，表示サイズが狭いと，表示されない領域ができるからです．

例えば，**Font**の設定のフィールドの右側には通常

〈図3-14〉 各種設定を行うノートブック



Edit ボタンがついています．また **Icon**，**PicturePath**，**HelpFile** などの設定フィールドの右側には **Find** ボタンがついています．これらはノートブックの幅を広げないと表示されません．

一方，高さが不十分であると，ページタグが一度に見渡せなかったり，ノートブックの上部が表示されないことがあります．

設定項目の意味がつかめないときには，その項目を操作して F1 キーを押すと関連するヘルプファイルが表示されます．いずれにしても，設定項目が多いので，少しずつ理解していきましょう．

ノートブックの右側についているのがタグです．次の設定を行います．

Color 頁の **BackColor** を **Cyan** に，**ForeColor** を **Blue** に，**Frame** 頁の **HideButton** と **MaximizeBotton** を選択，**Text** 頁の **Font** に対し，**Edit** ボタンで詳細メニューを開いて，**14.Times New Roman Bold Italic** を選択しました．

以上でプログラムが仕上がりました．再度，試行してみると，**図3-15** のように **Date** ボタンを押すと日付けが，また，**図3-2** に示したように，**Time** ボタンを押すと時刻が表示されます．

テストが終了すれば，コンパイルできます．具体的には，Ctrl+**X** キーを押すことにより，**EXE** 型ファイルの形で生成されます．生成されたプログラムはほかの機械でも実行できます．ただし，この **EXE** ファイル以外に実行時ライブラリ **VRObj.DLL** が，OS/2 環境の **LIBPATH** で指定されたパスに存在する必要があります．

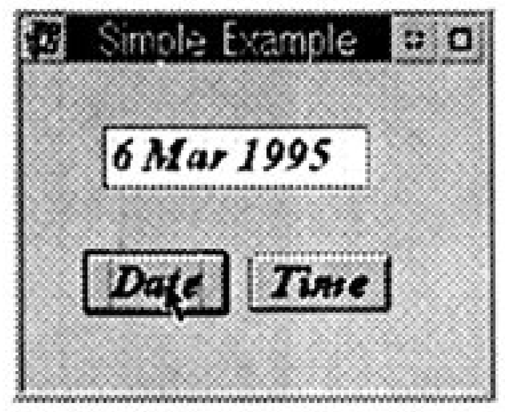
なお，VX-REXX の開発環境のメインウィンドウを閉じると，**VXR1.VRP** 以外に通常 **Window1.VRX** と **Window1.VRY** の二つが生成されます．後者は REXX のファイル，前者は補助的なバイナリファイルです．これらもファイル名を **VXR1** に合わせておいたほうが後々の管理がしやすいでしょう．そのため，一度，開発環境のメニューから **Project** の **Save as** を選択し，保管ファイル名として **VXR1** を選択して保管しておけば，以降はこれが，**.VRX**，**.VRY**，**.VRP** の共通名として扱われます．

この簡単な例題より VX-REXX におけるプログラミングの要点を理解していただけたでしょうか？

各オブジェクトについて，何かが起こった(Event)に対してサブルーチンが作られ，各オブジェクトにいかなる操作(action)を施すかを REXX で記述していきます．簡単な操作に対しては，構文が自動的に生成されます．また，裏方のメインルーチンも自動的に生成されるので，プログラマは，それらに注意を払う必要はありません．

この例で示されたように，オブジェクトの Cut，Copy，Paste，Delete を行うと，プロパティは継承され，Event ルーチンも生成，消滅されます．画面設計とコーディングの順序を適宜調整したほうがよいこともあります．

〈図3-15〉できあがったプログラムを実行する



〈図3-16〉両替計算プログラム



VX-REXX による例題(その 2)

最初の例題をやや詳しく説明しましたので，次の例題“両替計算”について重複する点は説明を省略し，**図3-16** に示すような画面設計が終わったところから始めます．

プロジェクト名は **VXR2.VRP** とします．VX-REXX は日本語版がまだ出ていませんが，図からもわかるように，英語版でも日本語の入力表示が可能です．ただし，**Font** の設定には日本語の表示が可能なものを選ぶことが必要です．

“日付”と表示しているところには，**DescriptiveText (DT_1)** を張り付けています．これを下方に等間隔で 4 個分複写しています．つまり **DT_1** を **Copy** で取り込んで **Paste** を 4 回続けて行います．その右側には同様にして作成した五つの **EntryField(EF_1～EF_5)** を並べています．なお，初期設定として表示位置は右揃えとし，**EF_4** には **95.0** の初期値を入力しています．

なお，**Tools** ウィンドウが不要になればこれをクローズすることもできます．必要ならば F7 キーを押せば再度表示できます．

次は，コーディングの番です．問題の設定として，

(1) プログラムを起動したとき

日付けと時刻を入力し，カーソルを **EF_3** に合わせる．

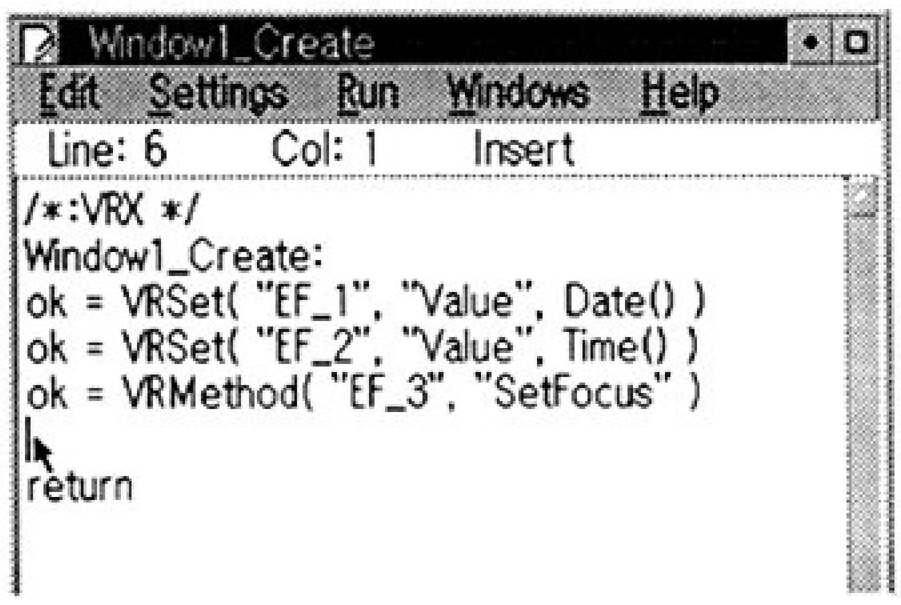
(2) **EF_3** の数値が変更されたとき

対応する円貨を **EF_5** に表示するものとする．

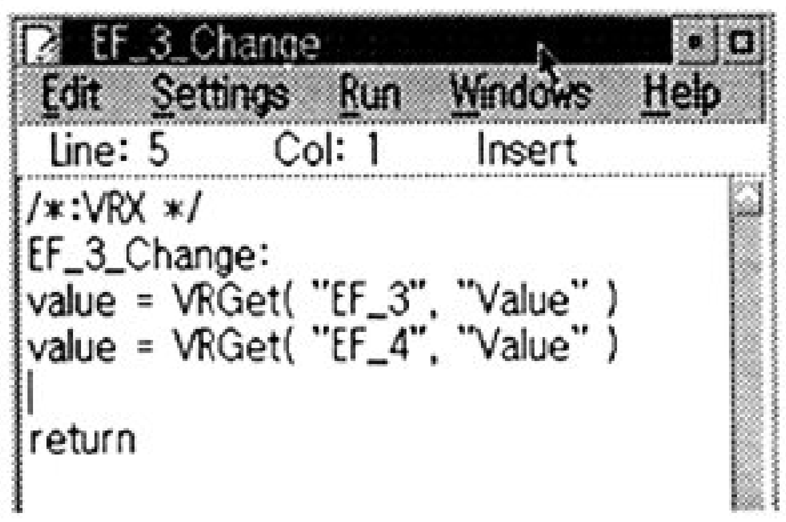
まず(1)ですが，**Window1** をマウスボタン 2 でクリックして，メニューより **Events**，**Create** を選択します．**Window1_Create** のスケルトンが現れます．日付け，時刻の入力は先の例題の変形ですので説明を省きます．

EF_3 のフィールドからスケルトンにドラッグ&ドロップすると，**EF_3：Select an action** ウィンドウが現れますので，**Methods** の **Set forcus to the object** を

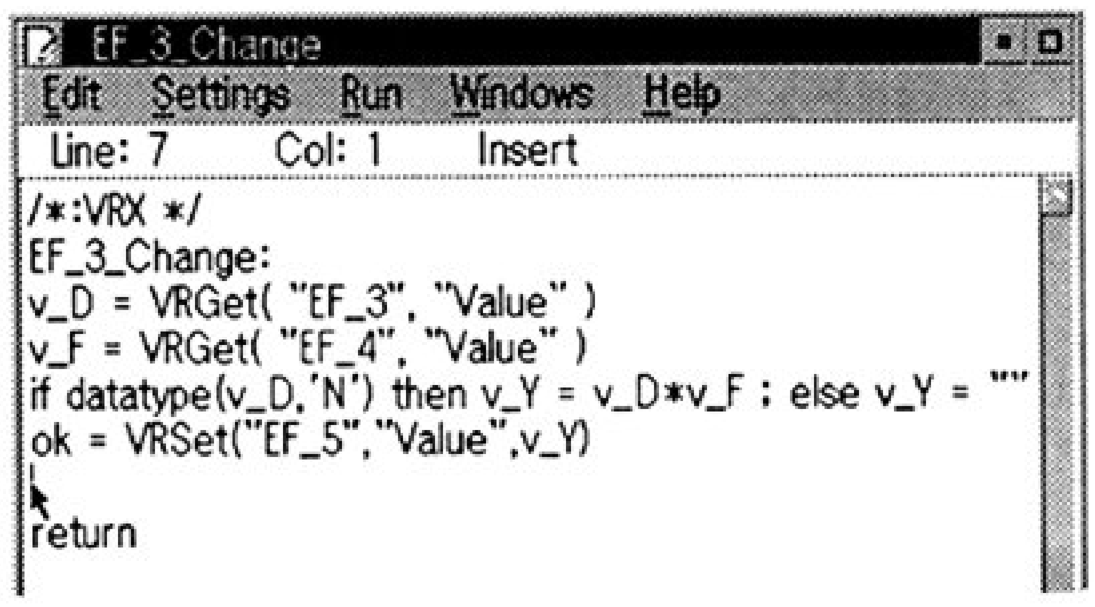
〈図3-17〉 コードが入った



〈図3-18〉 EF_3_Change のコードが入った



〈図3-19〉 計算式を記述する



〈リスト3-1〉 VXR2.VRX のコードの一部

```
/*:VRX         Main
*/
/*  Main
*/
Main:
/*  Process the arguments.
    Get the parent window.
*/
    parse source . calledAs .
    parent = ""
    argCount = arg()
    argOff = 0
    if( calledAs \= "COMMAND" )then do
        if argCount >= 1 then do
            parent = arg(1)
            argCount = argCount - 1
            argOff = 1
        end
    end; else do
        call VROptions 'ImplicitNames'
    end

  ・・・中略・・・

/*:VRX         EF_3_Change
*/
EF_3_Change:
v_D = VRGet( "EF_3", "Value" )
v_F = VRGet( "EF_4", "Value" )
if datatype(v_D,'N') then v_Y = v_D*v_F ; else v_Y = ""
ok = VRSet("EF_5","Value",v_Y)
return

  ・・・中略・・・

/*:VRX         Window1_Create
*/
Window1_Create:
ok = VRSet( "EF_1", "Value", Date() )
ok = VRSet( "EF_2", "Value", Time() )
ok = VRMethod( "EF_3", "SetFocus" )
return
```

選択します．**図3-17** のようにコーディングできました．

最後に(2)の条件をコーディングします．今度は，**EF_3** の **Events**，**Change** を選択します．現れたスケルトンにまず，**EF_3** から **Get Property** の **Value** を選択，**EF_4** からも同様にすると**図3-18** のようになります．

ここで，左辺をそれぞれ **v_D**，**v_F** と変更します．最後の操作は **EF_5** に **v_D** と **v_F** の積を書き込む手順となります．最終的に**図3-19** のように作成します．

ここで，**v_D** が数字でないときの判定が要点となります．また，**リスト3-1** に対応する **VXR2.VRX** の一部を示しておきます．簡単なプログラムの場合，REXX のコードは一つのファイルに作られ，変数の有効範囲や，サブルーチンの扱いなどは，一般の REXX と同じ文法が適用されます．

VX-REXX による例題(その 3)

次の例題は，**Timer** オブジェクトによる **"StopWatch"**のプログラム例を扱います．その完成時の姿を**図3-20** に示します．

一番上部に示したのは，ストップウォッチのディジタル表示部を模したエントリフィールド(**EF_1**)で，その下に三つのラジオボタンが並びます．また，画面からは隠してありますが，**Timer** オブジェクト(**TM_1**)を置いています．プロジェクト名は **VXR3.VRP** としています．

まず，この例題の問題設定をまとめておきます．

(1) プログラム起動時

表示を 0 で停止，**Start** ボタンのみ使用可．

(2) **Start** ボタンを押したとき

表示が 1 秒ごとにインクリメント，**Stop** ボタンのみ使用可．

(3) **Stop** ボタンを押したとき

表示は，その時の値で停止，**Start** あるいは **Reset** ボタンを使用可．

(4) **Reset** ボタンを押したとき

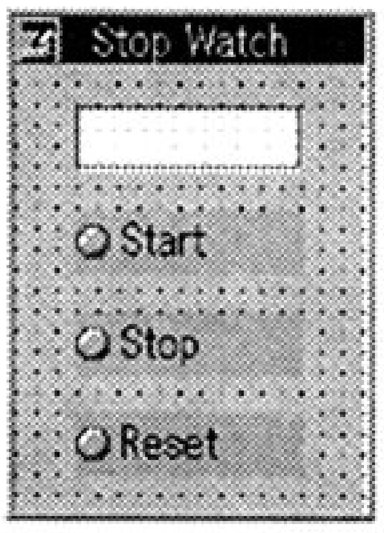
(1)と同様．

図3-21 に示したのは，画面設計の途中までと，**Window1_Create** ルーチンの制作途中です．これまでの例題と異なる点は，**v_T** なる変数を用いたこと，内部サブルーチン **Disp** および **SetStatus** の利用です．

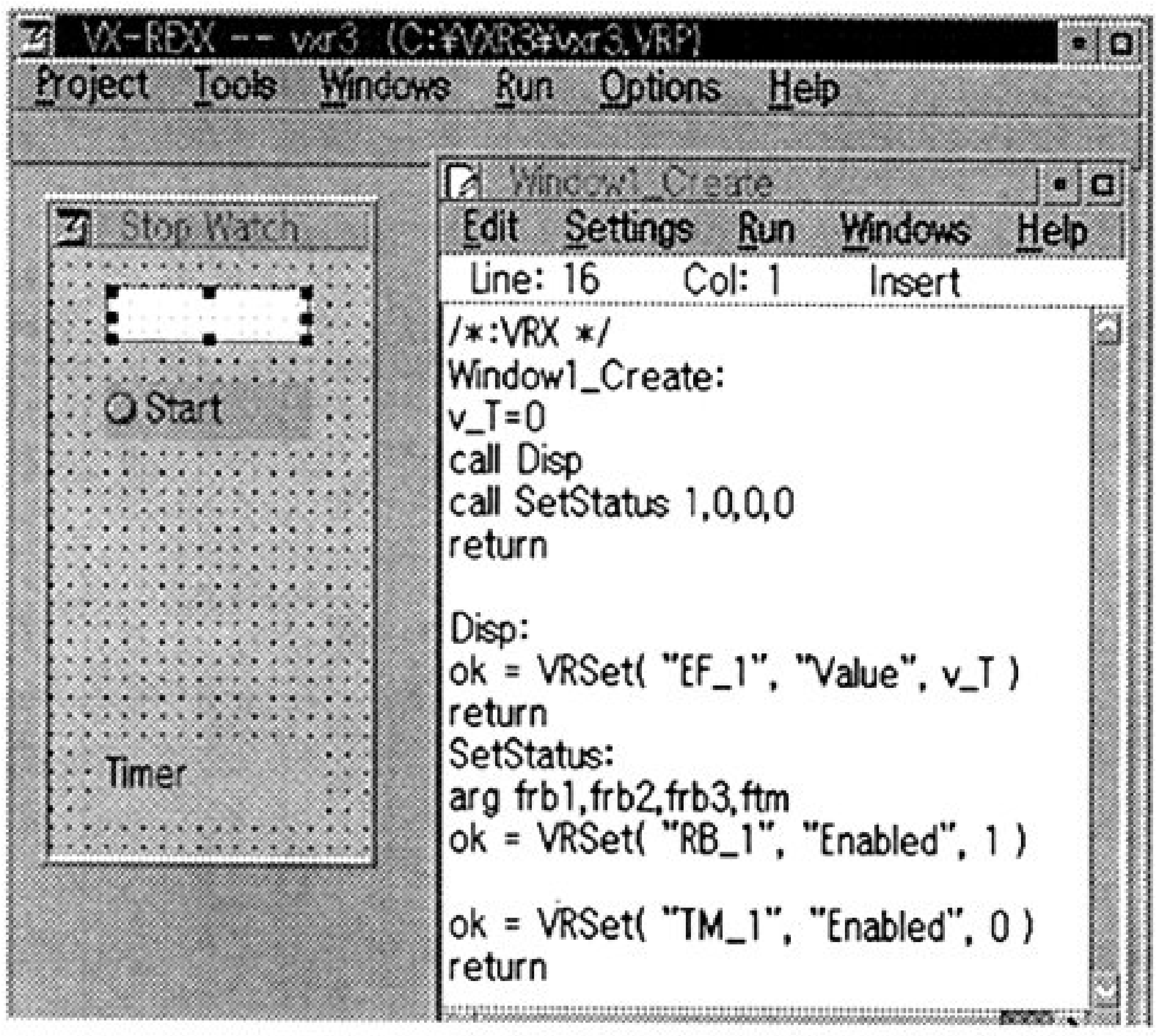
これらは，ほかのイベントに対応するルーチンからも利用できるので，プログラミングが簡潔に表現できます．

Window1_Create : での操作は，ストップウォッチの指示を **0** にして **Start** ボタン(**RB_1**)をイネーブル，**Stop** ボタン(**RB_2** を予約)，**Reset** ボタン(**RB_3** を予

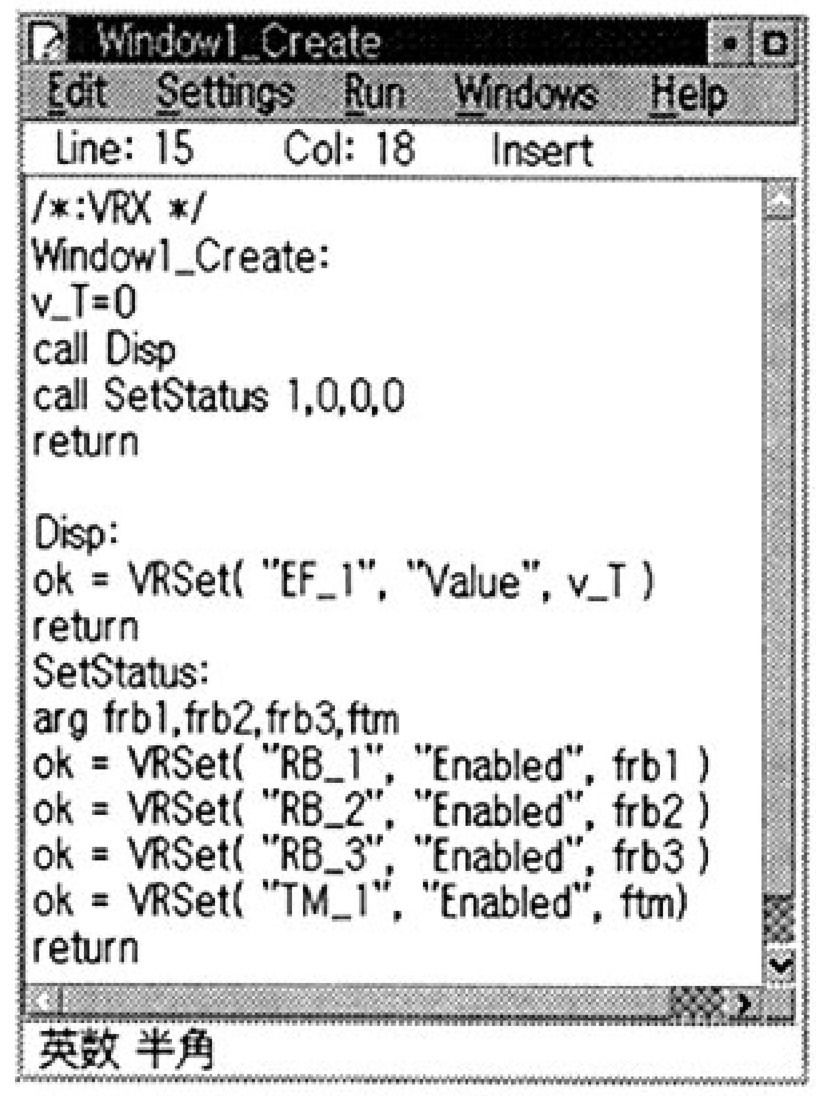
〈図3-20〉 ストップウォッチのプログラム



〈図3-21〉 ストップウォッチの制作途中



〈図3-22〉 SetStatus の完成



約) および **Timer (TM_1)** をディセーブルにすることです.

Disp: の内容は，これまでと同様にビジュアルプログラミングで自動的に生成できます．また，**SetStatus** の内容は未完成ですが，とりあえず **RB_1, TM_1** に対する操作を仮に作成しています.

これを元に REXX ソースを編集し，**SetStatus** を完成したものを**図3-22** に示します．また，**Start** ボタン (**RB_1**) を押したときの操作を**リスト 3-2** (a)に示します.

その後，**RB_1** をもとにコピー-ペースト-ペーストを行い，**RB_2** および **RB_3** を生成します．**RB_1_Click:** と同じ内容が **RB_2_Click:** および **RB_3_Click:** に作られますので，これを修正して完成します〔**リスト 3-2** (b)，**リスト 3-2** (c)〕.

最後に，**TM_1_Trigger:** の内容を**リスト 3-2** (d)に示します．**Delay** は 1000 ms つまり 1 秒に設定されていますので，1 秒ごとにこのルーチンが呼ばれます．**v_T** の値をインクリメントし，表示するという意味になります.

v_T やサブルーチンの利用あるいは，補助的コードの記述などで REXX に対する知識が必要となってきます.

VX-REXX による例題（その 4）

図3-23 に示した，今回の最後の例題 **Album** は典型的なビジュアルプログラミングで，コードはまったく書かずにでき上がります．プロジェクト名は **VXR4.VRP** です.

図3-24 に示したのは，**Window1** に **Notebook** オブジェクト (**NB_1**) を張り付けた状態です.

Window1 の設定は **Color** 頁の **BackColor** を **Cyan** に，**ForeColor** を **Blue** に，また，**Frame** 頁に **HideButton** および **MaximizeBotton** を設定，**Style** 頁から **LayoutStyle** として **SimpleDialog** を選択，また **Caption** を **Album** としています.

NB_1 の設定は **Color** 頁の **BackColor** を **Cyan** に，**Pages** 頁の **InitialPageList** を ;**SW_1+P1** ;**SW_2+P2** ;**SW_3+P3** として，また，**PreloadPages** を設定にしています．**～List** の設定では最初の記号がセパレータの役割をします．上記の **InitialPageList** の構文は（区切記号）（サブウィンドウ名）+（ページタグ名）の繰り返しとなります.

各頁はサブウィンドウの上に作ります．F8 キーを押すと**図3-25** に示した **Windows** リストの窓が現れます．最初は **Window1** のみでしょう.

ここで，ウィンドウから **New window** を選ぶと**図3-26** に示したような画面が現れます．最初は自動的に **SW_1** と名付けられます．**OK** を押すとこのサブウィンドウが生成されます．**SW_2, SW_3** も同様に生成できます.

次に，**SW_1** に **PictureBox** のオブジェクト (**Pict_1**) を張り付けます．設定としては，**Picture** 頁の **PicturePath** に，この例では **c:¥vxr4¥ex1.bmp**（これは例題 1 の実行画面を OS/2 のビットマップ形式でキャプチャしたもの）を設定，また **ResizePicture** を解除，また，**Style** 頁の **Border** を解除しています.

SW_2, SW_3 に同様の方法で画面設計してもよいのですが，**Pict_1** をもとにコピー-ペーストで作成 **PicturePath** を修正するほうが簡単です．サブウィンドウ間でもオブジェクトの複写が可能です.

以上でプログラミングは終了です．なお，サブウィンドウを生成しても，**Window1** と一緒に扱われますので，新たなファイルができるわけではありません.

VX-REXX のその他の機能

いくつかの例題で VX-REXX によるプログラミン

〈リスト3-2〉 VXR3.VRP プロジェクトのコードの一部

```
/*:VRX          RB_1_Click
*/
RB_1_Click:
call SetStatus 0,1,0,1
return
```

(a) Start ボタンを押したとき

```
/*:VRX          RB_2_Click
*/
RB_2_Click:
call SetStatus 1,0,1,0
return
```

(b) Stop ボタンを押したとき

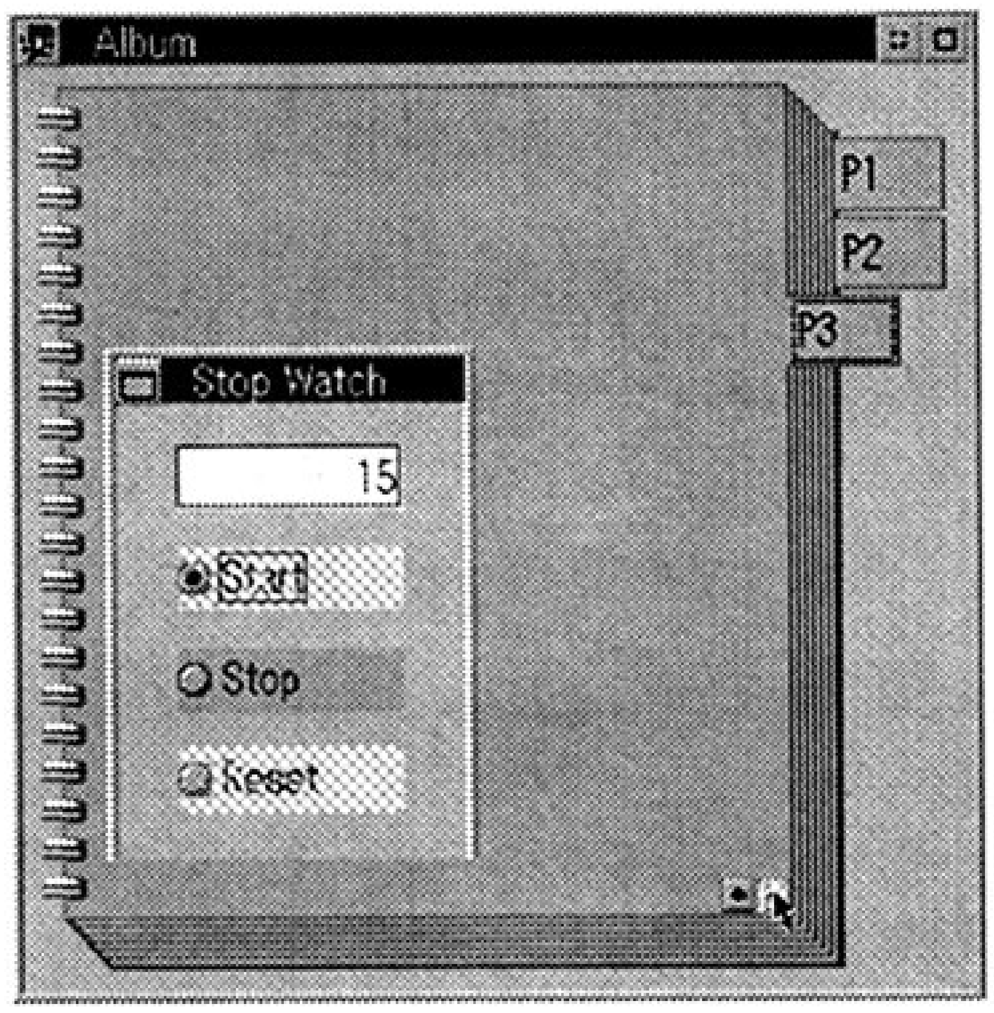
```
/*:VRX          RB_3_Click
*/
RB_3_Click:
v_T=0
call Disp
call SetStatus 1,0,0,0
return
```

(c) Reset ボタンを押したとき

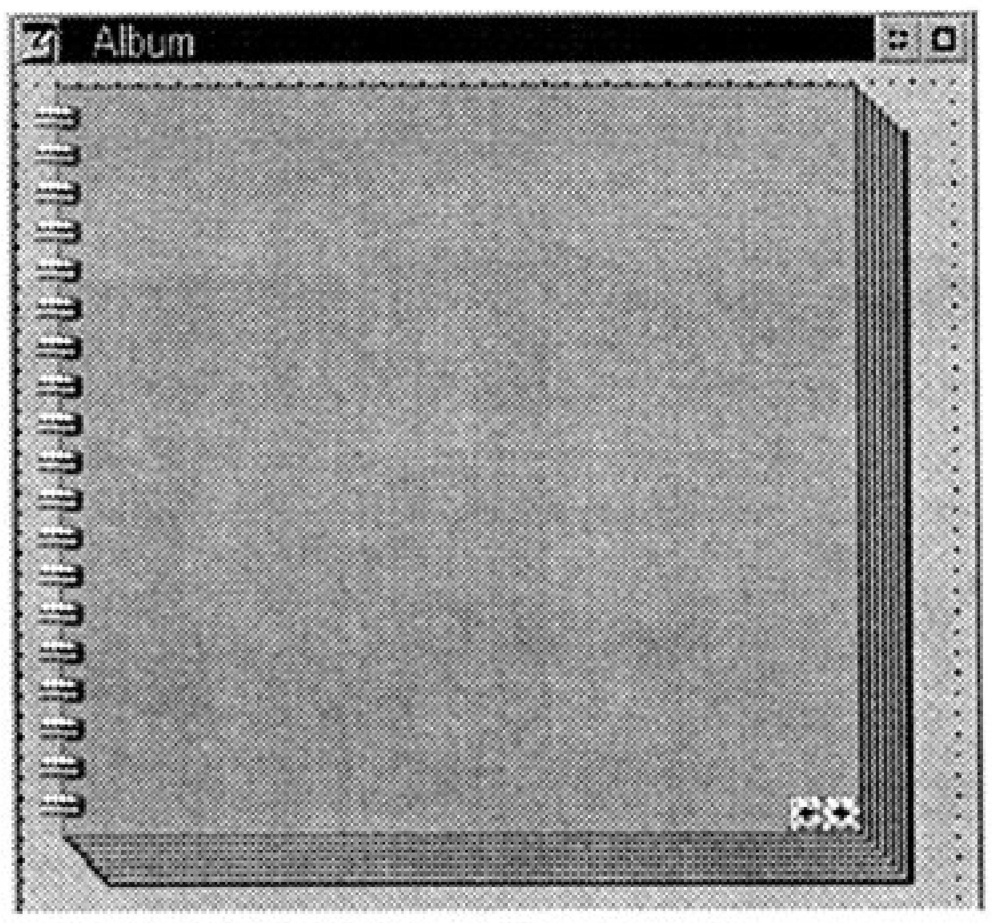
```
/*:VRX          TM_1_Trigger
*/
TM_1_Trigger:
v_t=v_T+1
call Disp
return
```

(d) **TM_1_Trigger** の内容

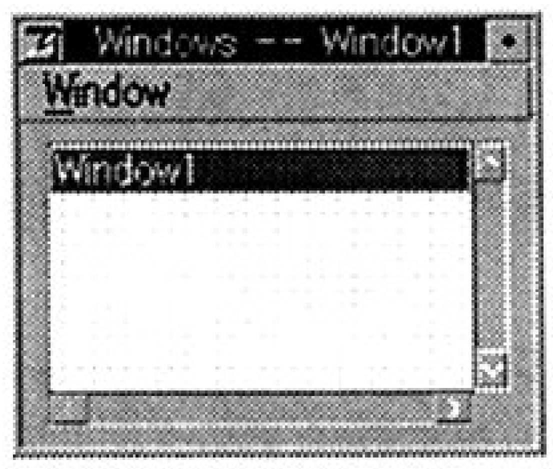
〈図3-23〉 プログラム例 Album



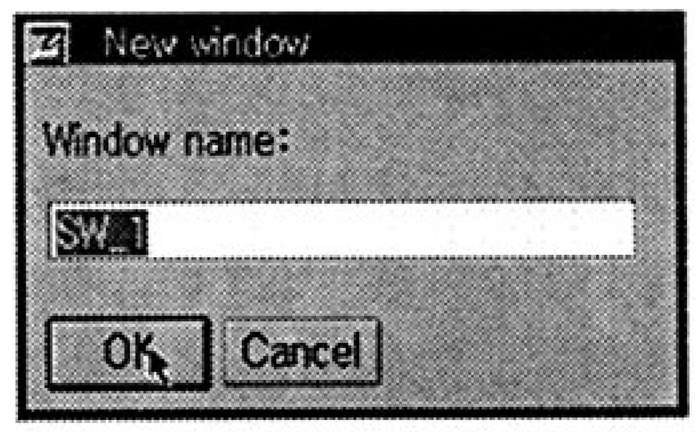
〈図3-24〉 Notebook オブジェクトを張り付けた



〈図3-25〉 ウィンドウズの表示



〈図3-26〉 Newwindow の名前が SW_1 とつく



グの一端を紹介しましたが，その機能はここで触れたもの以外に多くあります．概要を述べておきます．

(1) オブジェクト

その他の主なものとして **Container**，**DDEClient**，**Chart**，**Query** および **Connection** などがあります．特に最後の三つはクライアント-サーバー版のみに付属します．

(2) ドラッグ&ドロップ操作

ワークプレースシェルと同様の操作を応用プログラムに付加することができます．

(3) メニュー

通常の PM プログラムと同様に応用プログラムにメニューバー，プルダウンメニュー，ポップアップメニュー，さらに，カスケードメニューを仕込むことができます．

(4) 組み込みダイアログ

ファイルダイアログ，フォントダイアログ，メッセージダイアログ，プロンプトダイアログなどを使用することができます．

(5) データベースへのアクセス

特に，クライアント-サーバー版では，DB2/2 や ODBC 対応のデータベースを取り扱える(ただし，ODBC ドライバは別途用意する必要あり)．

(6) オブジェクトパック

Visual Basic の成功にサードパーティ製のコントロールパックの存在があるといわれています．VX-REXX においてもサードパーティ製オブジェクトパックとして DB Technologies 社の Entry Field Pack や Cirrus Technologies 社の Unite(image object set)などわずかですが製品が出つつあります．

次回は，本連載の最終回として VisPro/REXX を中心に述べたいと思います．

(注 1) IBM の従業員の開発したフリーウェアプログラムの意味．

(注 2) Compiler などで有名な Watcom は Power Soft の子会社，RDBMS の開発元 Sybase の孫会社．

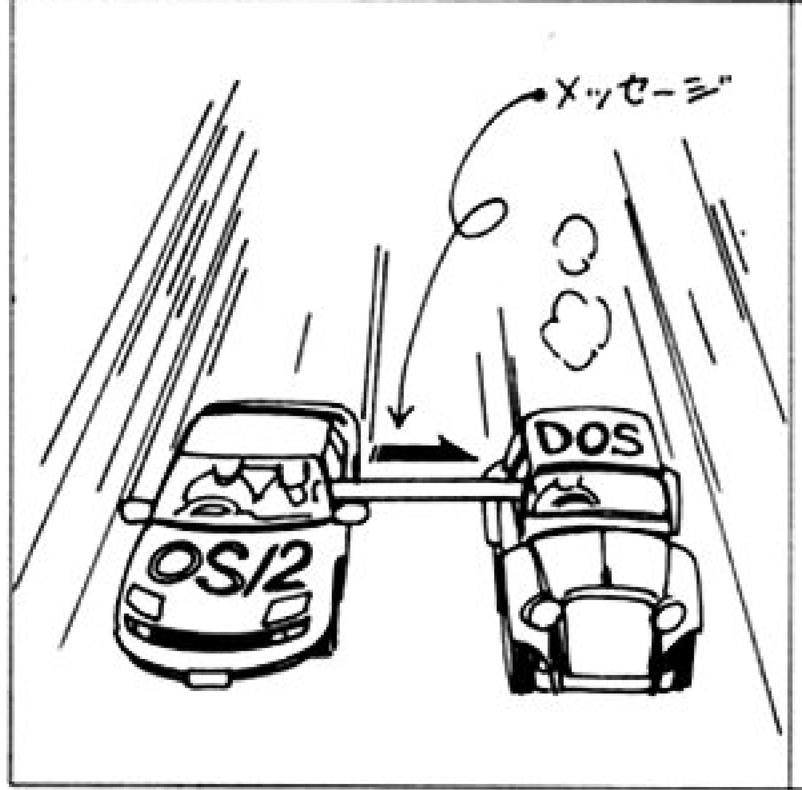
(注 3) V 2.1A Client/Server(英語)版による．

(注 4) この場合には **"Event info"** は現れない．

(注 5) VX-REXX は拡張子 **.BMP** の bitmap ファイル(OS/2 および MS-Windows)および拡張子 **.ICO** の icon ファイル(OS/2)を扱うことができる．

◆ 参考文献 ◆

(1) Watcom VX-REXX for OS/2 Programmer's Guide and Reference.
(2) Watcom VX-REXX Client/Server Edition Objects Guide.



GNU C/C++ Compilerでマスターする OS/2プログラミング

④ DOSも走る，OS/2ならいっしょに走る

岡田好一

前回では，マルチスレッド・プログラミングについて取り上げました．今回はマルチスレッド・プログラミングと並んでOS/2の大きな特徴の一つである名前付きパイプを使ったプロセス間通信について取り上げます．

● はじめに

マルチタスク・システムではプロセス間通信を当たり前のように使用します．

OS/2でもプロセス間通信を使用できますしいろいろな方法がありますが，中でも名前付きパイプは非常に強力な方法の一つです．

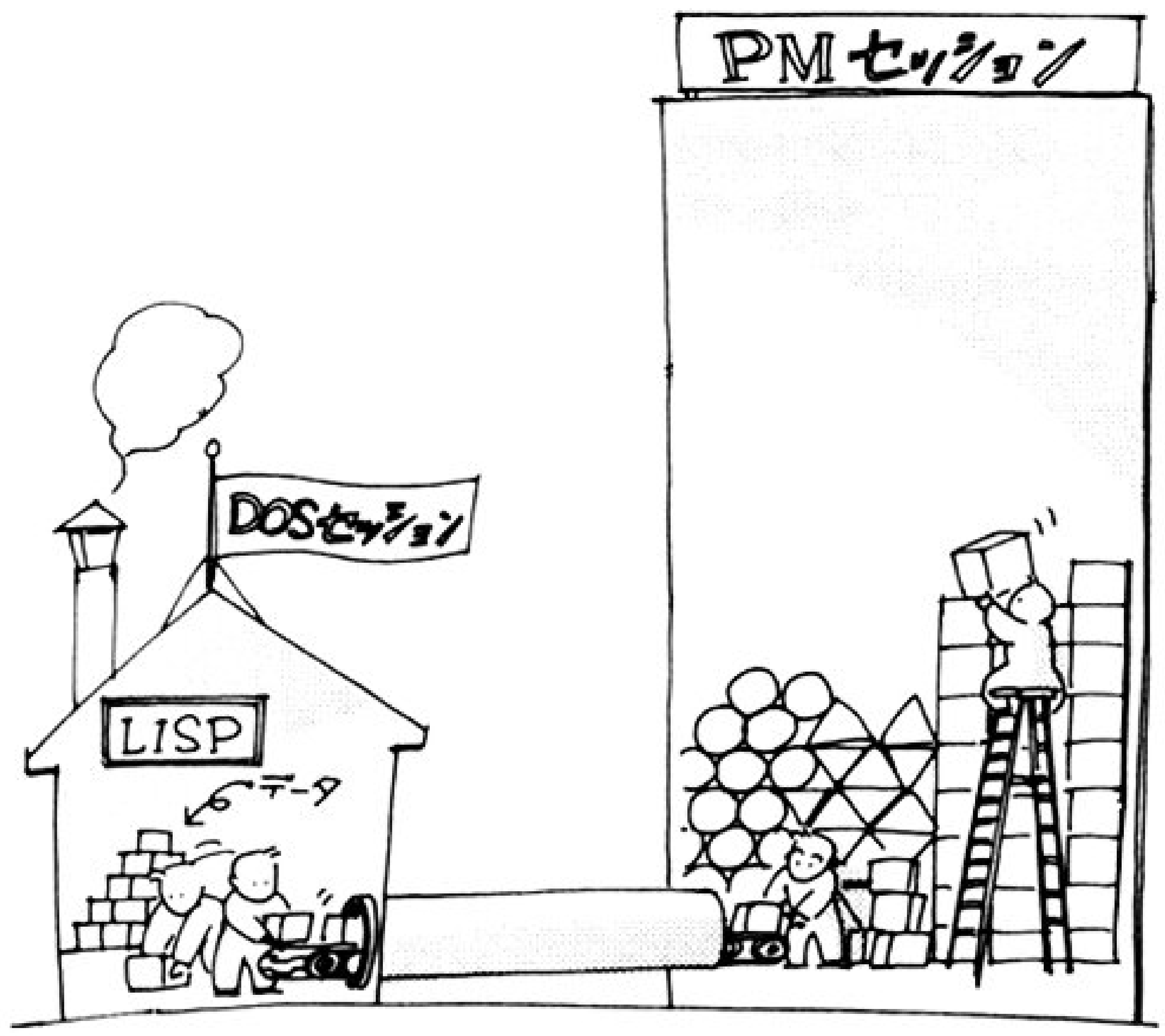
ここでは，名前付きパイプを使って，OS/2内のDOS/V上のプログラム(ここではLISPプログラム)を，コードの変更をすることもなく，グラフィック拡張するという例を紹介します．

アイデアとしては極めて単純です．パイプはプログラムからはファイルのように見えますから，DOS/Vプログラムからはファイルとのデータのやり取りが必要なだけです．

あとは，OS/2のプログラムが面倒を見てくれます．名前付きパイプに図形データを出力すると，別のOS/2のプロセスがそのデータを同時並行的に解析し，図形を表示します(**図4-1**)．

名前付きパイプというと何か複雑なものというイメージがあるようですが，本稿を読まれた後には，「そんなの当たり前」という感じになっているでしょう．

〈図4-1〉 名前付きパイプの動作



名前付きパイプについて

● サーバープロセス～名前付きパイプを生成～

OS/2の名前付きパイプの生成は，サーバーと呼ばれるプロセスが行います．つまり，サーバープロセスが名前付きパイプを生成し，接続の受け付けを開始します．

パイプの名前は，

¥pipe¥ttt

のように，**pipe**ディレクトリのファイルの形をしています．

しかし，**pipe**はディレクトリ名ではありません．ですから，**dir**コマンドで見ることは不可能です．

ちなみに，UNIXにも「名前付きパイプ」があります．しかし，UNIXではコマンドラインからでも名前付きパイプを作れますし，ディレクトリの一覧表示コマンド**ls**で見ることもできます．

また，適当なデスクトップシステムがあると，かわいいパイプアイコンが表示されたりもします．おもしろいことに，利用法はOS/2と同じく，クライアント/サーバー・プログラミングです．

● クライアントプロセス～サーバーの機能を利用～

サーバーの機能を利用するプロセスを，クライアントと呼びます．クライアントに必要なのは，名前付きパイプの名前を知っていることだけであり，通常のファイルのオープンと同様の方法でサーバーに接続できます．**DosOpen( )**を明示して使う必要はありません．

パイプが強力なのは，同期の問題がファイルの扱いに隠れてしまうことです．パイプのバッファが一杯になれば書き込み側が待たされ，パイプが空になると読み込み側が待たされます．これだけです．ほかに特別

〈リスト4-1〉名前付きパイプ，クライアント→サーバー(npipe1.c)

```
/* npipe1.c YOkada 950216 Named Pipe test */
/* client -> server(this) */

#define INCL_DOS
#include <os2.h>

const int LB = 1;

int main() {
  APIRET rc;
  static PSZ pipen="¥¥PIPE¥¥NPIPE1"; /* パイプの名前 */
  HFILE hfile;                       /* パイプのハンドル */
  char b[LB];
  ULONG n;

  rc = DosCreateNPipe(
    pipen,              /* パイプの名前 */
    &hfile,             /* パイプのハンドルへのポインタ */
    NP_ACCESS_DUPLEX,   /* 二重(双方向)パイプの指定 */
    1,                  /* パイプのインスタンスの数 */
    1000,               /* バッファの大きさ S→C */
    1000,               /* バッファの大きさ C→S */
    0);                 /* 省略時の待ち時間 */

  rc = DosConnectNPipe(hfile); /* クライアントが接続するのを待つ */

  for (;;) {
    rc = DosRead(hfile, b, LB, &n);
    if (0 == n) break;
    rc = DosWrite(1, b, LB, &n); }
}
```

なしかけはありません．

それでいて，並行処理をしている二つのプロセスのつじつまが合ってしまいます．

名前付きパイプのテストプログラム

● クライアントからサーバーへ

最初にちょっとした実験をしてみましょう．

リスト4-1は，サーバープロセスがクライアントからの文字列を読み込んで，標準出力に表示するプログラムです(**図4-2**)．

このサーバープロセスを，例えばOS/2ウィンドウセッションで起動しておき，別のOS/2またはDOS/Vセッションから，

type ttt.txt > ¥pipe¥npipe1

などと書き込みます(**ttt.txt**は適当なテキストファイル)．

ネットワーク経由の場合，

¥¥server_name¥pipe¥npipe1

などとなります．

サーバープロセスのセッションのウィンドウに，クライアントから送った文字列が表示されるはずです．

ところで，OS/2のウィンドウを二つ作る方法ですが，Warp なら Launch Pad から簡単に生成できます．J2.11の場合，OS/2のウィンドウから，あらかじめ**start**コマンドで作成しておきます．

▶プログラム(**リスト4-1**)の解説

(1) **DosCreateNPipe( )**

DosCreateNPipe( )は，名前付きパイプを作るAPI

〈図4-2〉テストプログラムの実行画面



関数です．

第1引数(入力)は，パイプの名前を指定します．ここでは**¥PIPE¥NPIPE1**です．

第2引数(出力)は，パイプのハンドルへのポインタです．ここでは**&hfile**です．パイプはファイルのように見えるので，**DosRead( )**などでそのまま使うことができます．

第3引数(入力)は，オープンモードです．DOSからの入力の場合，**NP_ACCESS_INBOUND**(入力専用)ではなく，**NP_ACCESS_DUPLEX**〔二重(双方向)パイプ〕を指定します．

第4引数(入力)は，パイプモードです．本プログラムでは明示していませんが，パイプ(バッファ)が使えない時は単純に待ち，バイトストリームを使います．

1はパイプのインスタンスの数です．この数を増やすことにより，同じ名前のパイプをいくつでも作れるということです．

第5引数(入力)は，サーバーからクライアントにデータを送る時に使うバッファの大きさです．

第6引数(入力)は，クライアントからサーバーにデータを送る時に使うバッファの大きさです．

第7引数(入力)は，**DosWaitNPipe( )**を用いる時，省略時の待ち時間の指定に使います．本プログラムでは**DosWaitNPipe( )**を使いません．

(2) **DosConnectNPipe( )**と**for**ループ

パイプを作成したら，クライアントが接続するのを**DosConnectNPipe( )**で待ちます．

forループでは，パイプから1文字読んでは標準出力(1)に文字を出力しています．

クライアントがファイルを閉じる操作をして，パイプの接続を切ると，**DosRead( )**の第4引数は0を返します．

● サーバーからクライアントへ

リスト4-2は，サーバープロセスからクライアントに文字列を送ります．サーバーは，標準入力(0)からの

〈リスト4-2〉名前付きパイプ，サーバー→クライアント（**npipe2.c**）

```
/* npipe2.c YOkada 950216 Named Pipe test */
/* server(this) -> client */

#define INCL_DOS
#include <os2.h>

const int LB = 1;

int main() {
  APIRET rc;
  static PSZ pipen="\\PIPE\\NPIPE2";
  HFILE hfile;
  char b[LB];
  ULONG n;

  rc = DosCreateNPipe(
    pipen, &hfile,
    NP_ACCESS_OUTBOUND, /* 出力専用 */
    1,
    1000, 1000, 0);

  rc = DosConnectNPipe(hfile);

  for (;;) {
    rc = DosRead(0, b, LB, &n);
    if (0 == n) break;
    rc = DosWrite(hfile, b, LB, &n); }
}
```

入力なので，例えば，

　　npipe2 < ttt.txt

などとします．キーボードから直接打ち込んでもかまいません．

　クライアントのウィンドウでは，

　　type ¥pipe¥npipe2

などとします．サーバーからパイプ経由で送られた文字列を表示するはずです．

▶プログラム（**リスト4-2**）の解説

　DosCreate( ) の第3引数，つまりオープンモードで，**NP_ACCESS_OUTBOUND**（出力専用）を指定しています．

　for ループでは，標準入力から1文字読み込み，パイプに文字を送っています．

　ファイルの終わりになるか，または **Ctrl-Z** などの入力によって，このサーバープロセスは終了してしまいます．クライアントプロセスのパイプからの読み込みは，エラーとなります．

DOS/V プログラムの出力データを PM を使って図形表示する！

　リスト4-3，**リスト4-4** は，図形表示のサーバープロセスのプログラムです．名前付きパイプからの文字列を解析して，図形を表示します（**図4-3**）．

● プロトコルの定義

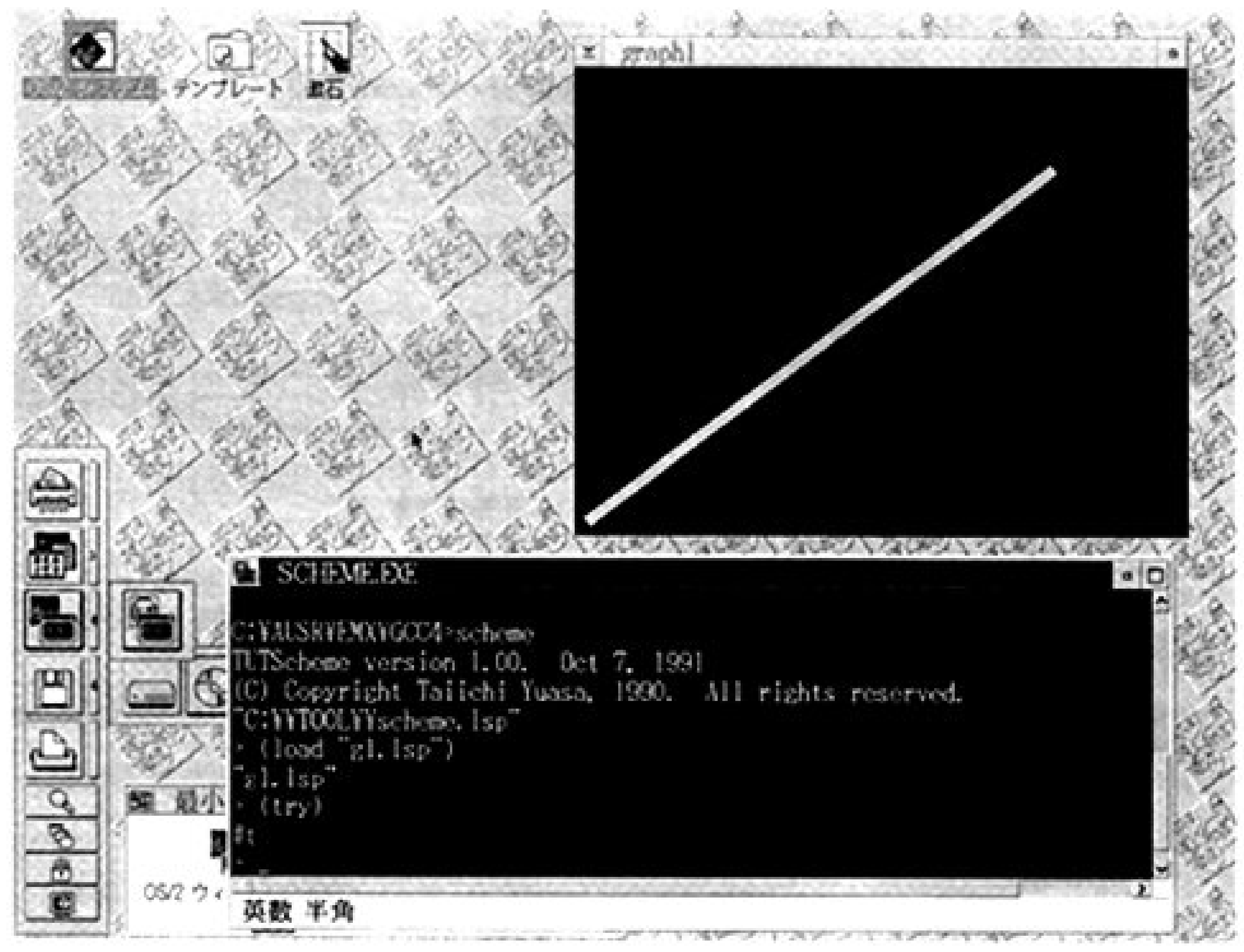
　まず，どのような文字列を受け取ったら図形を表示するかを定義します．グラフィック画面は，横512ドット，縦384ドットに固定します．

　リスト4-3 の **do_draw( )** 関数が，図形を描く部分です．内容は後ほど説明しますので **case** の行を見てください．各文字列の意味を**表4-1** に示します．

〈リスト4-4〉リスト4-3の定義ファイル（**graph1.def**）

```
NAME          GRAPH1 WINDOWAPI
DESCRIPTION   'graph1 YOkada 950322'
HEAPSIZE      65536
STACKSIZE     65536
EXPORTS       ClientWndProc
```

〈図4-3〉パイプを使った図形表示の実行例



　入力は1行単位です．例えば，

　　21 10 20 100 130

という行が入力されたら，(10，20)から(100，130)へ直線を描きます．最初の21が直線描画の合図ということです．

　WM_PAINT 処理を軽くするため，内部的なビットマップに描画し，必要に応じて，クライアント・ウィンドウに **GpiBitBlt( )** で内容を転送します．

　そのため，上記の直線描画は，内部的なビットマップに描画するのみです．

　ですから，図形を表示するには，

　　1

という行を明示的に送ります．

　画面の消去は，

　　2

で，色は黒のみです．消去は，すぐにウィンドウにも反映されます．

　図形は，直線のほかに長方形と楕円があります．長方形と楕円には，輪郭のみと塗りつぶしの2種類があります．

　　22 x0 y0 x1 y1

といった感じです．

　22 が塗りつぶし長方形，**23** が輪郭だけの長方形，**24** が塗りつぶしの楕円，**25** が輪郭だけの楕円です．楕円の場合，**x0 < x1**，**y0 < y1** とする必要があります．

　　10 r g b

で，直線の輪郭の色を指定します．**r**，**g**，**b** はそれぞれ赤，緑，青で，0～255のスケールで指定します．

〈表4-1〉
サーバーに送る文字列(プロトコル)の定義

1	表示
2	画面消去
10 r g b	直線/輪郭の色(r：赤, g：緑, b：青). それぞれ, 0~255 のスケールで指定
11 w	直線/輪郭の大きさ. w：太さ
15 r g b	塗りつぶしの色(r：赤, g：緑, b：青). それぞれ, 0~255 のスケールで指定
21 x0 y0 x1 y1	直線. (x0, y0)から(x1, y1)
22 x0 y0 x1 y1	塗りつぶし長方形. (x0, y0), (x1, y1)
23 x0 y0 x1 y1	輪郭のみの長方形. (x0, y0), (x1, y1)
24 x0 y0 x1 y1	塗りつぶしの楕円. (x0, y0), (x1, y1). ただし, x0<x1, y0<y1
25 x0 y0 x1 y1	輪郭のみの楕円. (x0, y0), (x1, y1). ただし, x0<x1, y0<y1

11 w

は, 直線の輪郭の太さの指定です.

塗りつぶしの色は,

15 r g b

で指定します.

テストするには, **graph1** を OS/2 セッションから起動しておいて, DOS または OS/2 セッションから,

type con > ¥pipe¥graph1

などとして, 上記の行を打ち込みます. 終了には, クライアントのセッションで, **Ctrl-Z** などとします.

● プログラム(リスト4-3)の概略

本プログラムは, PM メッセージを処理しながらパイプからの入力も監視しなければなりません. そのため, パイプ入力専用のスレッドを作ります.

● プログラム(リスト4-3)の説明 ～main( ) 関数～

まず, 画面の大きさを固定したウィンドウを作成します. 名前付きパイプ

¥pipe¥graph1

を **DosCreateNPipe( )** で生成します.

▶ビットマップの作成

メモリ・デバイスコンテキスト(デバイスに依存する情報)を **DevOpenDC( )** で作ります. プリンタやビットマップに対し描画するには, この関数でデバイスコンテキストを作る必要があります. ビットマップを用いるため, 第2引数に **OD_MEMORY** を指定しています. 第4引数以降は設定用の引数で, 今回は使いません.

描画の直接の対象となる表示空間は, **GpiCreate-**

3つの表示空間

OS/2 の GPI 関数は, 第1引数に表示空間(Presentation Space, PS)を指定します(例外はある). つまり GPI 関数は,「表示空間に描画する」というイメージとなります.

表示空間は, 装置独立, つまり, ウィンドウでもメモリ(ビットマップ)でもプリンタでも同様にグラフィックするための工夫です.

内部的には, カラーやスタイルなどのデータを保持する構造体です. HPS はその構造体の ID 番号というわけです.

表示空間にはキャッシュマイクロ表示空間, マイクロ表示空間, 通常(normal)表示空間の3種類があります.

● キャッシュマイクロ表示空間

キャッシュマイクロ表示空間は, **WinBeginPaint( )** や **WinGetPS( )** を普通に使うと得られる表示空間です. 対象デバイスはディスプレイに限られます.

システムが管理する領域を使うのでちょっと使うだけなら効率がよく, 表示をころころ変える必要がある場合に使います.

キャッシュマイクロ表示空間は, メッセージへの応答がすんだら, 速やかに **WinReleasePS( )** または **WinEndPaint( )** で離さないといけません. その際, 変更した属性は, デフォルトに戻ってしまいます.

● マイクロ表示空間

メモリやプリンタに表示する場合に必要です. **GpiCreatePS( )** 関数で作成します. 必ずデバイスコンテキスト(DC)と結び付け(associate)ます.

属性が長く保持できるので, ウィンドウが対象でも, フォントなどを扱う場合に便利です.

デバイスコンテキストは, 内部的には, 表示空間とデバイスドライバを結び付けるためのデータを保持する構造体です.

● 通常表示空間

マイクロ表示空間と同じく, **GpiCreatePS( )** で作成し, デバイスコンテキストと結び付けます.

マイクロ表示空間との違いは,

(1) 他のデバイスコンテキストに結び付きを変更できること

(2) セグメントと呼ばれる, 図形情報を保持する仕組みが使えること

です.

PS( ) で作ります．第 3 引数の表示サイズはビットマップの大きさを用いるので，ゼロにしておきます．**GpiCreatePS( )** の第 4 引数の意味は，**表4-2** のとおりです．

ウィンドウプロシージャの **WM_PAINT** で使っている **WinBeginPaint( )** は，キャッシュマイクロ表示空間を得ます．キャッシュマイクロ表示空間は高速ですが，制限も多いものです．

〈表4-2〉 GpiCreatPS( )の第 4 引数

PU_PELS	座標の単位をドット単位とする
GPIF_DEFAULT	内部的な座標の形式を 4 バイトに指定する
GPIT_MICRO	マイクロ表示空間を使う
GPIA_ASSOC	デバイスコンテキストと表示空間を結び付ける

ビットマップに描画する場合は，マイクロ表示空間，または通常表示空間を用いる必要があります．

ビットマップを作成するため，**BITMAPINFOHEADER2** 構造体に必要なデータを書き込み，**GpiCreateBitmap( )** を用いています．第 3 引数以降は，ビットマップを初期設定する時に使います．

GpiSetBitmap( ) は，メモリ・デバイスコンテキストにビットマップを結び付けます．

▶ カラーテーブルのモード選択

デフォルトでは，**GpiSetColor( )** で 16 色しか選ぶことはできません．256 色以上を表示できる計算機が大部分と思いますので，カラーテーブルを RGB モードに設定します．

GpiCreateLogColorTable( ) の第 3 引数に，**LCOLF_**

〈リスト4-3〉 図形表示，サーバープログラム(graph1.c)

```
/* graph1.c YOkada 950322 graphic expansion for DOS */

#define INCL_WIN
#define INCL_DOS
#define INCL_GPI
#define YOKO 512
#define TATE 384
#include <os2.h>
#include <string.h>

static HWND hFrame, hClient;
static HFILE hfile;
static HPS hpsm;

VOID APIENTRY thread2(HWND);
MRESULT EXPENTRY ClientWndProc(HWND, ULONG, MPARAM, MPARAM);

int main() {
  HAB hab;
  HMQ hmq;
  QMSG qmsg;
  static CHAR szClientClass[] = "GRAPH1";
  static ULONG lCFlag = FCF_TITLEBAR | FCF_SYSMENU |
   FCF_MINBUTTON | FCF_BORDER |
   FCF_SHELLPOSITION | FCF_TASKLIST;
  APIRET rc;
  RECTL rcl;
  BOOL fs;

  static PSZ pipen="¥¥PIPE¥¥GRAPH1";

  HDC hdcm;
  SIZEL sizel;
  BITMAPINFOHEADER2 bih2;
  HBITMAP hbm, hbm0;

  TID tid;

  hab = WinInitialize(0);
  hmq = WinCreateMsgQueue(hab, 0);
  WinRegisterClass(hab, szClientClass, ClientWndProc,
  CS_SIZEREDRAW | WS_ANIMATE, 0);
  hFrame = WinCreateStdWindow(
  HWND_DESKTOP, 0, &lCFlag, szClientClass,
  "graph1", 0, NULLHANDLE, 0, &hClient);

  WinSetRectEmpty(hab, &rcl);
  rcl.xRight = YOKO;
  rcl.yTop = TATE;
  WinCalcFrameRect(hFrame, &rcl, FALSE);


  fs = WinSetWindowPos(hFrame, NULLHANDLE, 0, 0,
     rcl.xRight - rcl.xLeft,
     rcl.yTop - rcl.yBottom,
     SWP_SIZE | SWP_SHOW);

  rc = DosCreateNPipe(
    pipen, &hfile,
    NP_ACCESS_DUPLEX,
    1,
    1000, 1000, 0);
/* ビットマップの作成 */
  hdcm = DevOpenDC(hab, OD_MEMORY, "*", 0, NULL, NULLHANDLE);

  sizel.cx = 0; sizel.cy = 0;
  hpsm = GpiCreatePS(hab, hdcm, &sizel,
    PU_PELS | GPIF_DEFAULT | GPIT_MICRO | GPIA_ASSOC);
  memset(&bih2, 0, sizeof(BITMAPINFOHEADER2));
  bih2.cbFix = sizeof(BITMAPINFOHEADER2);
  bih2.cx = YOKO; bih2.cy = TATE;
  bih2.cPlanes= 1;
  bih2.cBitCount = 8;
  hbm = GpiCreateBitmap(hpsm, &bih2, 0, NULL, NULL);
  hbm0 = GpiSetBitmap(hpsm, hbm);
/* カラーテーブルのモード選択 */
  GpiCreateLogColorTable(hpsm, LCOL_PURECOLOR, LCOLF_RGB, 0, 0, NULL);
/* スレッドの作成 */
  rc = DosCreateThread(&tid, (PFNTHREAD)thread2, hFrame, 0, 8192);

  while (WinGetMsg(hab, &qmsg, NULLHANDLE, 0, 0))
   WinDispatchMsg(hab, &qmsg);

  GpiDestroyPS(hpsm);
  DevCloseDC(hdcm);
  GpiDeleteBitmap(hbm);
  WinDestroyWindow(hFrame);
  WinDestroyMsgQueue(hmq);
  WinTerminate(hab);
  return 0;
}
/* ウィンドウプロシージャ */
MRESULT EXPENTRY
ClientWndProc(HWND hwnd, ULONG msg, MPARAM mp1, MPARAM mp2) {
  HPS hps;
  static POINTL ptls[4] =
  {0, 0, YOKO, TATE, 0, 0, YOKO, TATE};

  switch (msg) {
  case WM_PAINT:
    hps = WinBeginPaint(hwnd, NULLHANDLE, NULL);
    GpiBitBlt(hps, hpsm, 3, ptls, ROP_SRCCOPY, BBO_OR);
    WinEndPaint(hps);
    return 0;
  }
  return WinDefWindowProc(hwnd, msg, mp1, mp2);
}

/* thread2 第2スレッド */

const int LB = 1;
const int LS = 80;

void do_atol(unsigned char **s, int *v) { /* atol()に似た動作をする */
  int st, fg;

  *v = 0; fg = 0; st = 0;
  for (;;) {
    if (**s == '¥0') break;
    switch (fg) {
    case 0:
      if (**s == '-') {
  st = 1;
  fg = 1; }
      else if (('0' <= **s) && (**s <= '9')) {
  *v = **s - '0';
  fg = 1; }
      break;
    case 1:
      if (('0' <= **s) && (**s <= '9')) {
  *v *= 10; *v += **s - '0'; }
```

RGB を指定します．ディザによる混色を防ぐには，第2引数に **LCOL_PURECOLOR** を選択します．256 色表示の計算機で純色を指定すると，赤 7 階調，緑 8 階調，青 4 階調の 224 色が使われるようです．

GpiCreateLogColorTable( ) の第 4 引数以降は，カラーを間接指定する時に使います．

なお，パレットを使いたい方は，**GpiCreatePalette( )** などを調べてください．

RGB モードでは，赤緑青，それぞれ 0～255 の階調指定ができ，**GpiSetColor( )** の第 2 引数に，

r * 65536 + g * 256 + b

の数値を指定します．

▶スレッドの作成

パイプを監視する，**thread2( )** スレッドを **DosCreateThread( )** を用いて開始します．

▶ウィンドウプロシージャ

本プログラムのウィンドウプロシージャ，**ClientWndProc( )** は，**WM_PAINT** メッセージに応答するだけです．他のメッセージは，**WinDefWindowProc( )** が処理します．

WM_PAINT への応答では，**GpiBitBlt( )** を用いて，ビットマップのデータをウィンドウに転送します．**GpiBitBlt( )** 関数は拡大縮小が可能で，元の絵との相互作用を細かく調節できる強力な関数です．

● プログラム(リスト4-3)の説明～第 2 スレッド～

第 2 スレッドは，パイプからの入力を監視しています．1 行入力されたら，ビットマップに描画したり，第 1 スレッドに，ビットマップからウィンドウへの転送

〈リスト4-3〉 図形表示，サーバープログラム(graph1.c，つづき)

```
      else {
   fg = 2; }
      break;
    }
    if (fg == 2) break;
    (*s)++; }
  if (st == 1) *v = -*v;
}

const int MV = 5;

void do_sscan(unsigned char *s, int *v) {
  int i;

  for (i = 0; i < MV; i++) {
    do_atol(&s, v++); }
  return;
}

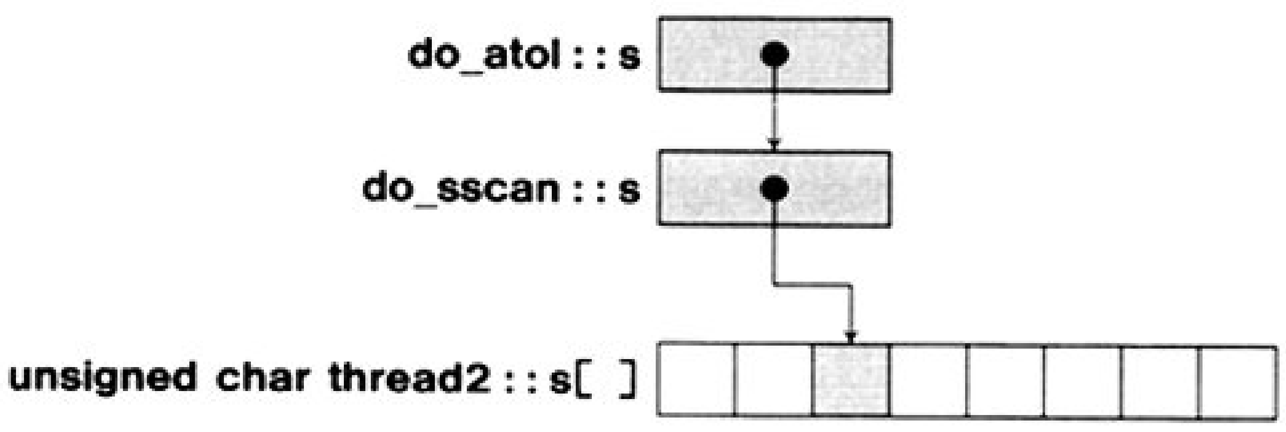
void do_draw(int *v) {
  POINTL ptl;
  static LONG clbk = 0, cleg = 16777215, clfil = 8421504;

  switch (v[0]) {
  case 1: /* flush */
    WinInvalidateRect(hClient, NULL, FALSE);
    break;
  case 2: /* clear screen */
    GpiSetColor(hpsm, clbk);
    ptl.x = 0; ptl.y = 0;
    GpiMove(hpsm, &ptl);
    ptl.x = YOKO - 1; ptl.y = TATE - 1;
    GpiBox(hpsm, DRO_FILL, &ptl, 0, 0);
    WinInvalidateRect(hClient, NULL, FALSE);
    break;
  case 10: /* edge color */
    cleg = 65536 * v[1] + 256 * v[2] + v[3];
    break;
  case 11: /* edge width */
    GpiSetLineWidthGeom(hpsm, v[1]);
  case 15: /* fill color */
    clfil = 65536 * v[1] + 256 * v[2] + v[3];
    break;
  case 21: /* line */
    GpiSetColor(hpsm, cleg);
    GpiBeginPath(hpsm, 1);
      ptl.x = v[1]; ptl.y = v[2];
      GpiMove(hpsm, &ptl);
      ptl.x = v[3]; ptl.y = v[4];
      GpiLine(hpsm, &ptl);
    GpiEndPath(hpsm);
    GpiStrokePath(hpsm, 1, 0);
    break;
  case 22: /* rectangle fill */
    GpiSetColor(hpsm, clfil);
    GpiBeginPath(hpsm, 1);
      ptl.x = v[1]; ptl.y = v[2];
      GpiMove(hpsm, &ptl);
      ptl.x = v[3]; ptl.y = v[4];
      GpiBox(hpsm, DRO_OUTLINE, &ptl, 0, 0);
    GpiEndPath(hpsm);
    GpiFillPath(hpsm, 1, FPATH_ALTERNATE);
    break;
  case 23: /* rectangle edge */
    GpiSetColor(hpsm, cleg);
    GpiBeginPath(hpsm, 1);
      ptl.x = v[1]; ptl.y = v[2];
      GpiMove(hpsm, &ptl);
      ptl.x = v[3]; ptl.y = v[4];
      GpiBox(hpsm, DRO_OUTLINE, &ptl, 0, 0);
    GpiEndPath(hpsm);
    GpiStrokePath(hpsm, 1, 0);
    break;
  case 24: /* ellipse fill */
    GpiSetColor(hpsm, clfil);
    GpiBeginPath(hpsm, 1);
      ptl.x = v[1]; ptl.y = v[2];
      GpiMove(hpsm, &ptl);
      ptl.x = v[3]; ptl.y = v[4];
      GpiBox(hpsm, DRO_OUTLINE, &ptl, v[3] - v[1], v[4] - v[2]);
    GpiEndPath(hpsm);
    GpiFillPath(hpsm, 1, FPATH_ALTERNATE);
    break;
  case 25: /* ellipse edge */
    GpiSetColor(hpsm, cleg);
    GpiBeginPath(hpsm, 1);
      ptl.x = v[1]; ptl.y = v[2];
      GpiMove(hpsm, &ptl);
      ptl.x = v[3]; ptl.y = v[4];
      GpiBox(hpsm, DRO_OUTLINE, &ptl, v[3] - v[1], v[4] - v[2]);
    GpiEndPath(hpsm);
    GpiStrokePath(hpsm, 1, 0);
    break;
  }
}

VOID APIENTRY
thread2(HWND hFrame) {
  APIRET rc;
  UCHAR b[LB];
  ULONG ulRead;
  unsigned char s[LS + 1];
  int si;
  BOOL fs;
  POINTL ptl;
  int v[MV];

  rc = DosConnectNPipe(hfile);
  GpiSetColor(hpsm, 255);
  ptl.x = 0; ptl.y = 0;
  GpiMove(hpsm, &ptl);
  ptl.x = YOKO - 1; ptl.y = TATE - 1;
  GpiBox(hpsm, DRO_FILL, &ptl, 0, 0);
  WinInvalidateRect(hClient, NULL, FALSE);
  si = 0; s[0] = '¥0';
  for (;;) {
    rc = DosRead(hfile, b, LB, &ulRead);
    if (ulRead == 0) break;
    if (b[0] == '¥n') {
      s[si] = '¥0';
      do_sscan(s, v);
      if (v[0] != 0) do_draw(v);
      si = 0; s[0] = '¥0'; }
    else if (b[0] == '¥t') {
      if (si < LS) s[si++] = ' '; }
    else {
      if (si < LS) s[si++] = b[0]; } }
  fs = WinPostMsg(hFrame, WM_CLOSE, NULL, NULL);
}
```

〈図4-4〉 do_atol( )の引数s



〈表4-3〉 GpiEndPath( )の後の関数

GpiStrokePath( )	太さのある線や輪郭を描画する
GpiFillPath( )	境界および内部を塗りつぶす
GpiOutlinePath( )	細い線で線や輪郭を描画する
GpiModifyPath( )	太さのある線や輪郭の，輪郭を得る
GpiPathToRegion( )	領域(ハンドルで扱える)を定義する
GpiSetClipPath( )	クリップ領域を定義する

を指示します．

▶ 文字列→数値変換関数 **do_atol( )**

do_atol( ) は，Cライブラリの **atol( )** に似た動作をする関数です．数字の一塊を検出し，数値を得て，＊**v** にセットします．符号を考慮します．

仮引数の **s** は，**unsigned char** へのポインタへのポインタです．**do_atol( )** の **s** が指すのは，**do_sscan( )** の仮引数の **s** です．**do_sscan( )** の仮引数の **s** は，**thread2( )** で定義されている **s[ ]** の要素を指します(**図4-4**)．

do_atol の仮引数の **v** は，**do_sscan( )** の仮引数の **v** のコピーです．

▶ 文字列検出関数 **do_sscan( )**

Cのライブラリの **sscanf( )** にほんの少し似ているので，**do_sscan( )** と名付けました．文字列から五つの数字の塊を検出し，数値を **v[ ]** に代入します．数字が検出できなければ0が代入されます．

▶ 描画関数 **do_draw( )**

第1スレッドに **WM_PAINT** を出すため，**WinInvalidateRect( )** 関数を用いています．

カラーの指示では，**static** 変数に **GpiSetColor( )** で用いる数値を代入しているだけです．直接使わない理由は，直線も輪郭も塗りつぶしの色も，すべて **GpiSetColor( )** でセットするからです．

線の太さは，**GpiSetLineWidthGeom( )** で指定しています．線の太さをいろいろ変更するには，パスを使わなければなりません．パスの線の太さを決めるのが，**GpiSetLineWidthGeom( )** です．

▶ パスの指定

パスは，**GpiBeginPath( )** と **GpiEndPath( )** の間に囲まれた部分の図形で指定します．

ごく簡単には，**GpiMove( )** といくつかの **GpiLine( )** で作成された折れ線を考えてください．つまり，両端のある折れ線や，閉じた折れ線図形の集合です．

アウトラインフォントの輪郭も利用できます．この座標の移動情報は，**GpiEndPath( )** の後の関数(**表4-3**)で使います．

▶ 第2スレッドのメインルーチン **thread2( )**

まず，**DosConnectNPipe( )** で，クライアントの接続を待ちます．

パイプが接続されたら，確認(デバッグ)のためにビットマップを青で塗りつぶして，第1スレッドに表示させています．

DosRead( ) に特別なことはありません．ファイルから読むのと同じ操作です．

クライアントがパイプの端を閉じると，**for** ループが終了し，第1スレッドに **WM_CLOSE** メッセージを送り(post)ます．

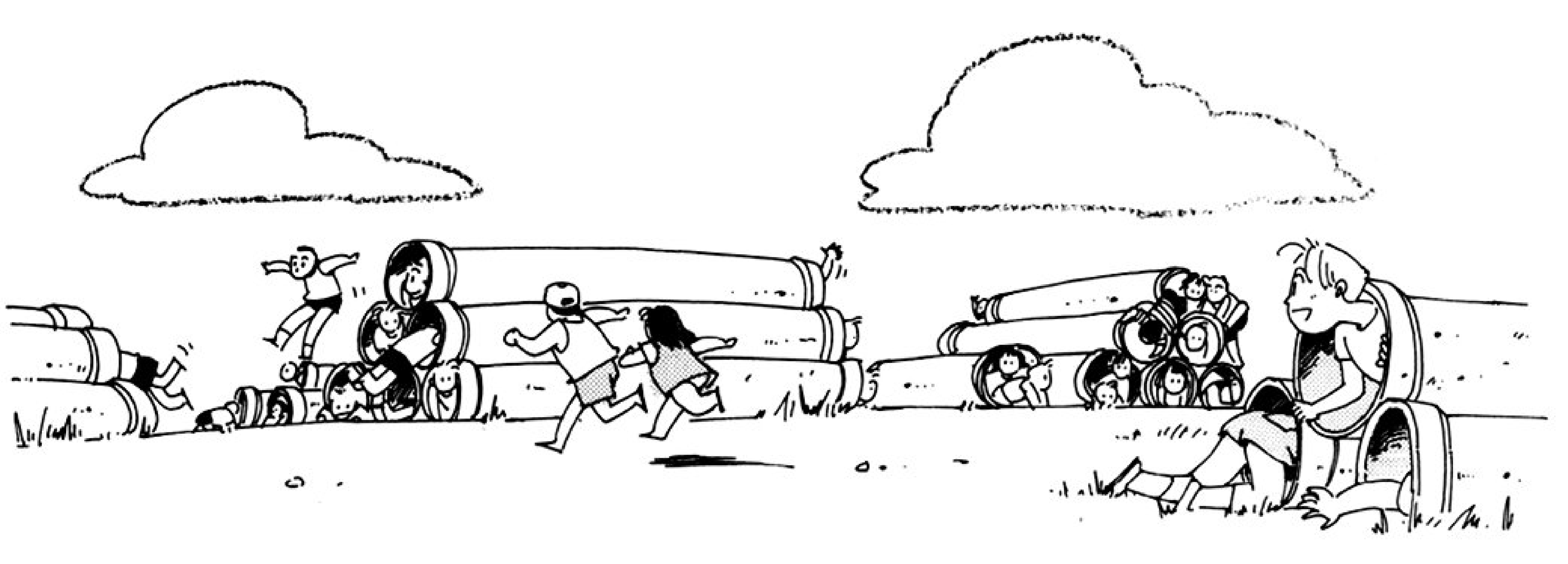
▶ **graph1.def** ファイル(**リスト4-4**)

PMプログラミングなので，リンカーへの指示のために **.def** ファイルを定義します．

コンパイルは，

```
gcc -o graph1.exe graph1.c graph1.def
                              -Zomf -Zmtd -s
```

などとします．



● LISP側の対応

LISP側の対応としては，

(gline 100 100 300 200)

といった関数が使われた場合に，**¥pipe¥graph1**に対して，

21 100 100 300 200

という文字列を送るだけです．

ライブラリ**g1.lsp**を用意します(**リスト4-5**)．といっても，エディタで**g1.lsp**というテキストファイルを作るだけです．

LISPといっても，ほとんど初歩のBASIC的なことしかしていないので，解読は容易でしょう．

● プログラムの使い方

サーバープロセス**graph1**をOS/2セッションから起動します．黒い画面のウィンドウが現れますから，画面の適当な位置に置いておきます．

DOS/Vウィンドウセッションを開始します．

c:¥>scheme

などとして，Scheme(最近よく使われているLISPの一方言)を起動します．Schemeがクライアントということです．

>(load "g1.lsp")

として，ライブラリを読み込みます．お試し関数**try**を使ってみましょう．

> (try)

黒い画面に，薄い赤の直線が描画されるはずです．

なお，テストに用いたLISPは，岩波ソフトウェアライブラリScheme(TUTScheme)です．

まとめ
～ほかのOSで同じことができるか？～

プロセス間通信と言っても，なんだか当たり前のことに思えるでしょう．

LISPは，ファイルのつもりで名前付きパイプに図形情報を書き込んでいるだけです．

サーバープロセスも，**DosRead( )**で図形情報を読んで，適当にビットマップやウィンドウに書き込んでいるだけです．

筆者も，あまりにあっけなく接続できたので，拍子抜けしたくらいです．

でも，ほかのOSでできるでしょうか？ もし，DOS/V(PC-DOS)のみで同じことを行うとすれば，どのようなプログラミングになるのでしょうか．

ほかのウィンドウシステム(Windows 3.1など)を使っていたとしたら，どのようにしてPC-DOSのプログラムと連携するのでしょうか．

PC-DOSのLISPシステムのコードに，いっさい手を付けずに，TSRもなしで，ネットワーク機能やGUI機能が追加できるのです．

OS/2は夢と魔法のOSと言ってよいでしょう．

〈リスト4-5〉図形表示クライアント，LISPプログラム(g1.lsp)

```
; g1.lsp YOkada 950324 graphic extension test

(define (ginit)
  (set! fil (open-output-file "¥¥pipe¥¥graph1")) )

(define (gflush)
  (display 1 fil) (newline fil) )

(define (gcls)
  (display 2 fil) (newline fil) )

(define (glcolor r g b)
  (display 10 fil) (write-char #¥space fil)
  (display r fil) (write-char #¥space fil)
  (display g fil) (write-char #¥space fil)
  (display b fil) (newline fil) )

(define (glwidth w)
  (display 11 fil) (write-char #¥space fil)
  (display w fil) (newline fil) )

(define (gfcolor r g b)
  (display 15 fil) (write-char #¥space fil)
  (display r fil) (write-char #¥space fil)
  (display g fil) (write-char #¥space fil)
  (display b fil) (newline fil) )

(define (gline x0 y0 x1 y1)
  (display 21 fil) (write-char #¥space fil)
  (display x0 fil) (write-char #¥space fil)
  (display y0 fil) (write-char #¥space fil)
  (display x1 fil) (write-char #¥space fil)
  (display y1 fil) (newline fil) )

(define (gfrect x0 y0 x1 y1)
  (display 22 fil) (write-char #¥space fil)
  (display x0 fil) (write-char #¥space fil)
  (display y0 fil) (write-char #¥space fil)
  (display x1 fil) (write-char #¥space fil)
  (display y1 fil) (newline fil) )
(define (grect x0 y0 x1 y1)
  (display 23 fil) (write-char #¥space fil)
  (display x0 fil) (write-char #¥space fil)
  (display y0 fil) (write-char #¥space fil)
  (display x1 fil) (write-char #¥space fil)
  (display y1 fil) (newline fil) )

(define (gfell x0 y0 x1 y1)
  (display 24 fil) (write-char #¥space fil)
  (display x0 fil) (write-char #¥space fil)

  (display y0 fil) (write-char #¥space fil)
  (display x1 fil) (write-char #¥space fil)
  (display y1 fil) (newline fil) )

(define (gell x0 y0 x1 y1)
  (display 25 fil) (write-char #¥space fil)
  (display x0 fil) (write-char #¥space fil)
  (display y0 fil) (write-char #¥space fil)
  (display x1 fil) (write-char #¥space fil)
  (display y1 fil) (newline fil) )

(define (try)
  (ginit) (gcls)
  (glcolor 255 128 128)
  (glwidth 10)
  (gline 10 10 400 300)
  (gflush) )

(define (try2)
  (ginit) (gcls)
  (try2a 0) (gflush) )

(define (try2a n)
  (cond ((= n 256) #t)
    (#t
      (gfcolor 0 0 n)
      (gfrect (* n 2) 0 (1+ (* n 2)) 383)
      (try2a (1+ n)) )))
```

Appendix　OS/2 純正コンパイラ/開発環境，IBM CSet＋＋について

この連載では，筆者があるとき「GNU で OS/2 するなんて簡単さ」と言い切ってしまったので，GNU C/C＋＋を使っています．GNU C/C＋＋は魅力のある処理系ですが，それまでは，IBM CSet＋＋を使っていました．

IBM CSet＋＋は，ひと言でいえば，大人の気分になれる開発環境です．

IBM CSet＋＋パッケージは，「IBM C/C＋＋ツール」，「ワークフレーム/2」，「OS/2 ツールキット」という三つの開発ツールをセットしたものです．

● C/C＋＋ツール(C/C＋＋コンパイラ)

32 ビット，フルセットの ANSI C/C＋＋コンパイラです．わざわざ「フルセット」などと言うのは，テンプレートや例外処理がきちんと規格書通り組み込まれているからです．

最適化スイッチも豊富で，ずいぶん前から Pentium コードにも対応していました．

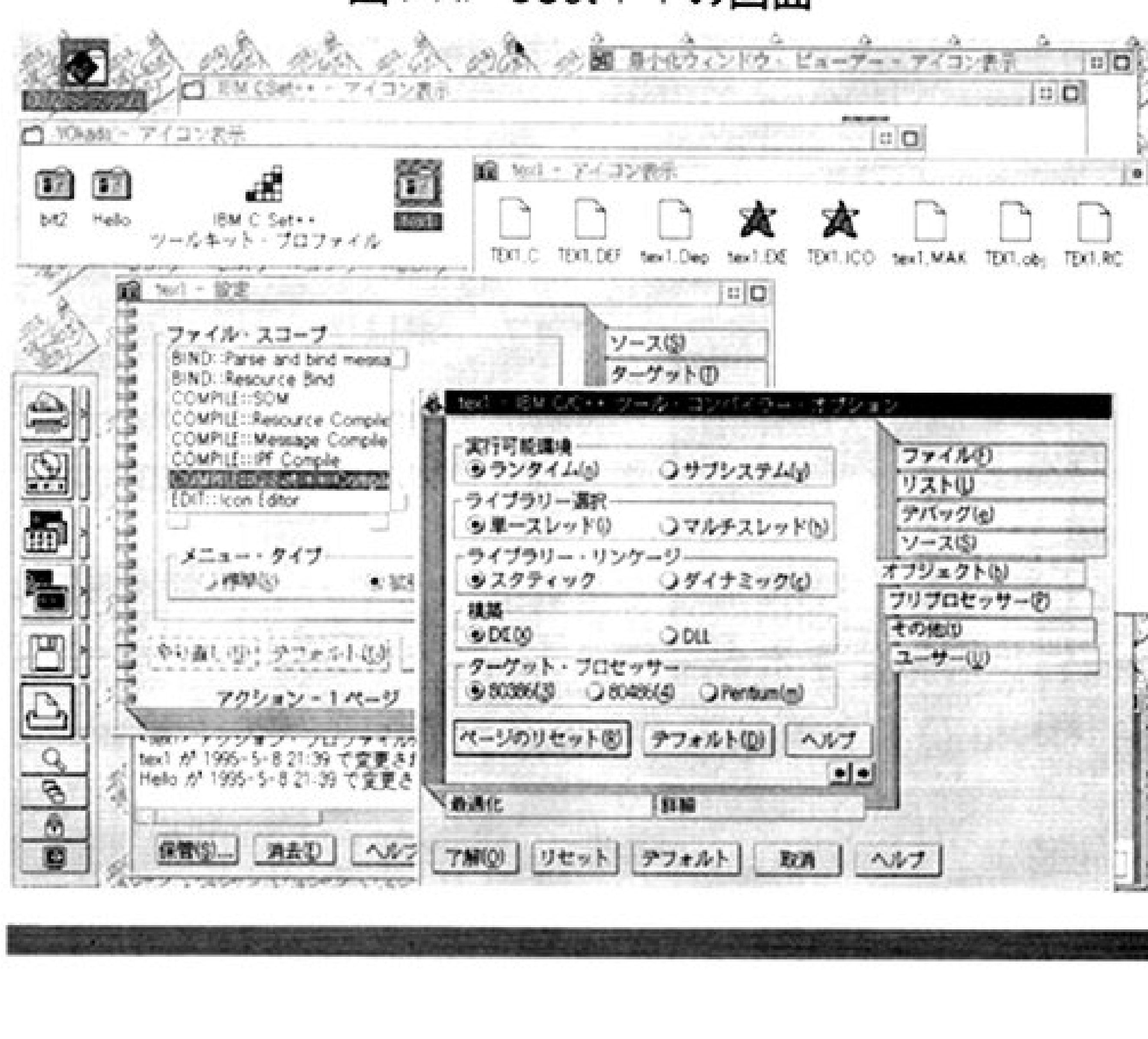
クラスライブラリも充実しており，他のパソコンコンパイラによくある，ユーザーインターフェース・クラスライブラリとコレクション・クラスライブラリのほか，ワークステーション系の C＋＋コンパイラに付いてくる，C＋＋スタンダード・クラスライブラリもあります．

● ワークフレーム/2(開発環境)

ワークフレームと呼ばれる開発環境が，2 種類入っています．

IBM ワークフレーム/2 バージョン J 1.1 は，よくある MDI 風のインターフェースを持ちます．

〈図4-A〉 CSet＋＋の画面



バージョン J 2.1 は，OS/2 のワークプレースシェルをそのまま使います．

両者の操作性は異なりますが，内容はよく似ています(J 2.1 は大規模な開発にも耐えられるよう設計されています)．

プロジェクトを定義し，用意しておいたソースファイルを指定します．

MakeMake と呼ばれるユーティリティを使い，make ファイルを作成します．つまり，make ファイルの作り方や，コンパイラやリンカのオプションを詳しく知らなくてもコンパイルができます．自動生成された make ファイルを手で編集してもかまいません．

その make ファイルを用いて，ソースファイルから **.exe** や **.dll** を作成します．

これらの操作が，GUI でできます．

● OS/2 ツールキット

ツールキットには，OS/2 の API 関数を使うためのヘッダーファイルや，ドキュメント，ダイアログエディタなどのユーティリティ，多くのサンプルプログラムが入っています．ヘルプファイルを作る，IPF コンパイラも含まれています．

英語版ですが，サンプルが豊富な MMPM/2 用のツールキットも付属しています．

● CSet＋＋の操作感

ワークステーションや大型機の経験があると，パソコンのコンパイラはおもちゃっぽくて，と感じることがあると思います．

しかし，この IBM CSet＋＋は，どちらかというと，ワークステーションのコンパイラを連想させます．

有名な Windows 3.1 のコンパイラ/開発環境もよくできていて，特に不満があるわけでもないのに，CSet＋＋を一度経験してしまうと，できれば OS/2 でコンパイルしたい，という気分になってしまいます．

その原因一つは「安心して 32 ビットできる」ということと，コンパイラにたのもしさが感じられる，という点にあるのだと思います．

さらに OS/2 のマルチタスクの爽快感が加わります．

このような「開発時の痛快な気分」というのは大切だと思うのですが，いかがでしょうか．

驚くのは価格です．ワークステーションの言語処理系は 30～40 万円が相場ですが，この CSet＋＋は，パソコンコンパイラの価格で買えます．

もし，余裕があるのなら，一段とよい気分でコンパイルするための投資をされてはいかがでしょうか．

Excel 5.0のVBA活用マニュアル

第7回　セル範囲の書式

西田雅昭

前回は，セル範囲の書式設定に関して **NumberFormat** プロパティだけを説明しましたが，セルの中の書式に関しては，ほかにもまだいろいろな指定ができます．

セル範囲の書式

セル範囲の書式設定を行う際に最も簡便なのは，**AutoFormat** メソッドを使うことです．これを利用すると，表示形式，フォント，配置，罫線，パターンを一度に設定することができます．

AutoFormat メソッドの書式は，

```
Range.AutoFormat(format,number,font,
                 alignment,border,pattern,width)
```

です．名前付き引数 **format** に書式を指定します．書式はExcelに組み込まれていて，**表7-1** のように19通りあります．

この引数の指定を省略した場合には，**xlClassic1** になります．これだけでは何のことか分からないかもしれません．

ワークシートを選択して適当な表を作り，表全体を選択してから，「書式」→「オートフォーマット」を選択してください．「オートフォーマット」ダイアログボックス(**図7-1**)を表示します．

ここで，「テーブル書式」リストボックスにある書式名がこの **format** なのです．「サンプル」ピクチャボックスで完成した書式を見ることができますから，これを覚えておいてください．

では，実際にやってみましょう．**sample** ディレクトリにある **sales.els** を使って説明します．効果が分かるように，「ツール」→「オプション」→「表示」パネルで，「枠線」チェックボックスのチェックを取って，枠線を表示しないようにしてください(**図7-2**)．

続いて，コードペインで，次のように入力します．

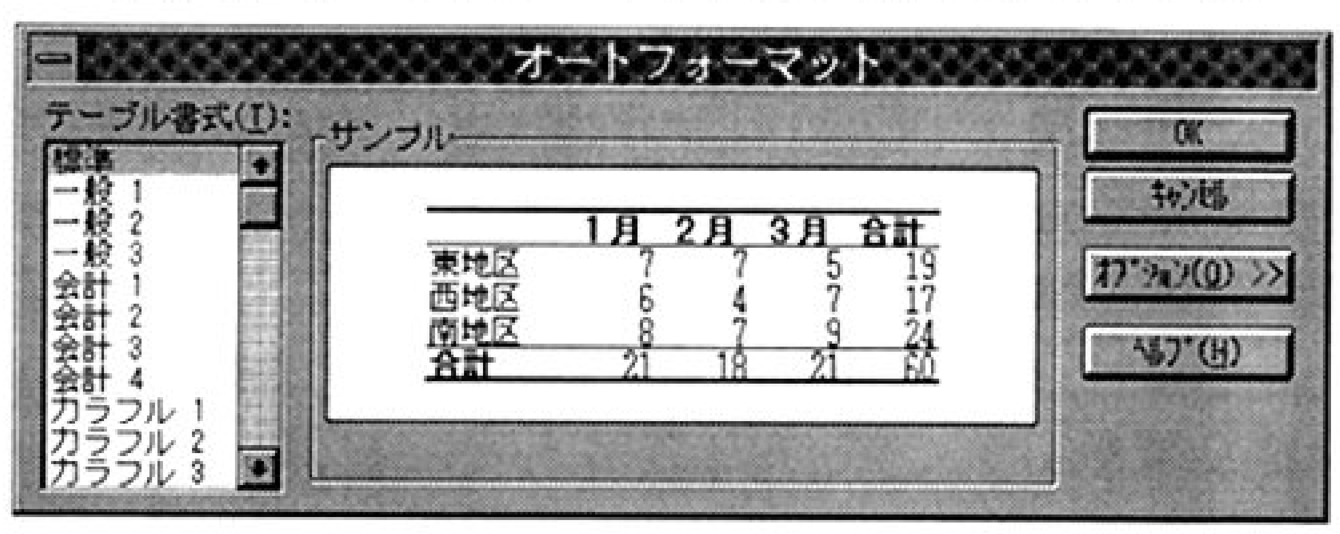
```
Range("B4 : D8").Select
Selection.AutoFormat format := xl3DEffects2
```

セル範囲を選択しているので，反転していてよく分かりません(**図7-3**)が，ワークシートに戻り[注1]，他のセルを選択すれば美しい姿を見ることができます(**図7-**

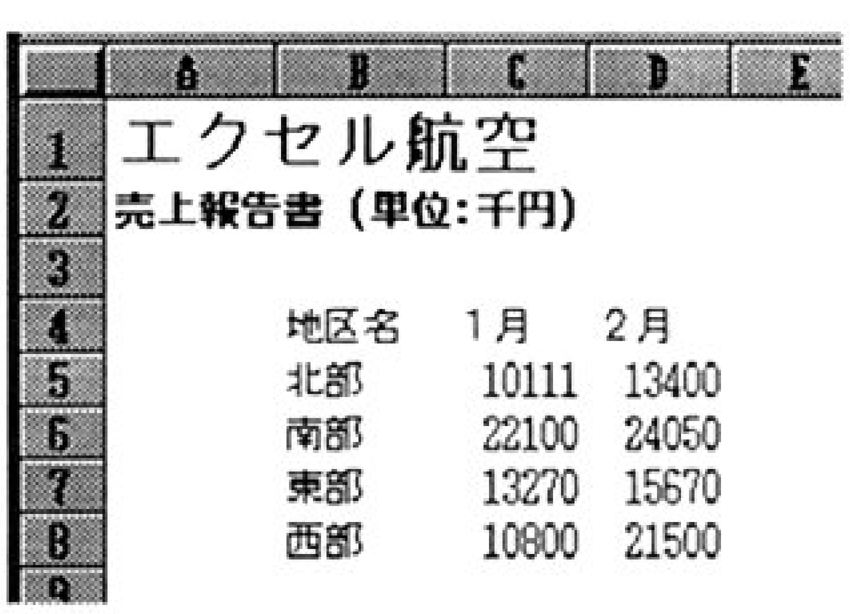
〈表7-1〉組み込み format の書式

xlClassic1	一般 1	xlList1	リスト 1
xlClassic2	一般 1	xlList2	リスト 1
xlClassic3	一般 1	xlList3	リスト 1
xlAccounting1	会計 1	xl3DEffects1	3-D1
xlAccounting2	会計 1	xl3DEffects2	3-D1
xlAccounting3	会計 1	xlLocalFormat1	表 1
xlAccounting4	会計 1	xlLocalFormat2	表 1
xlColor1	カラフル 1	xlSimple	標準
xlColor2	カラフル 1	xlNone	書式削除
xlColor3	カラフル 1		

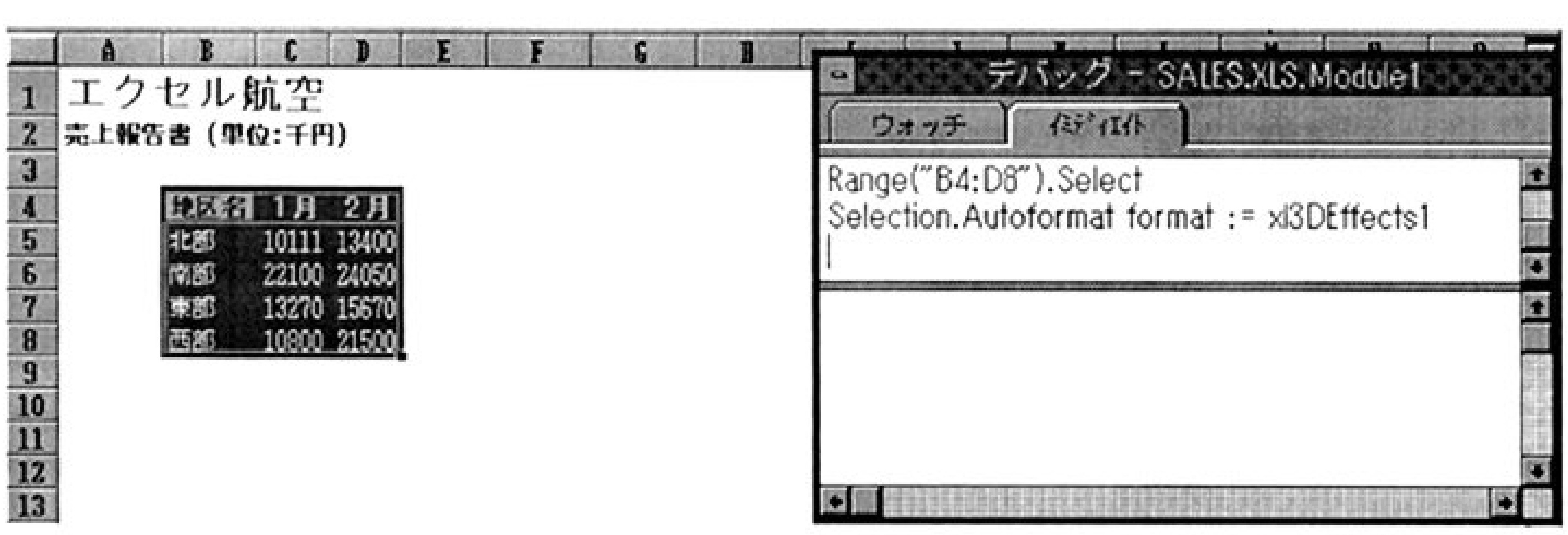
〈図7-1〉オートフォーマットのダイアログボックス



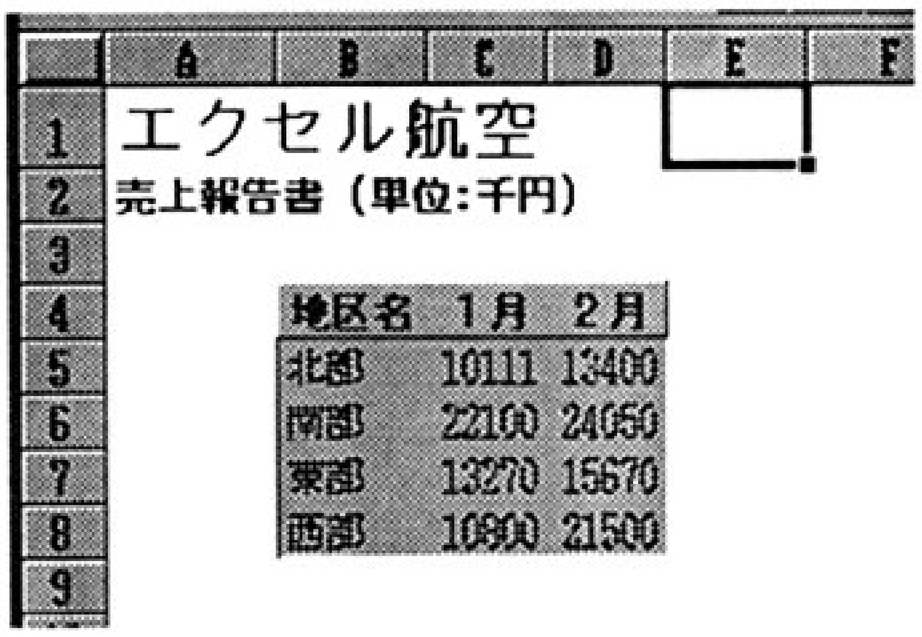
〈図7-2〉ワークシートの例



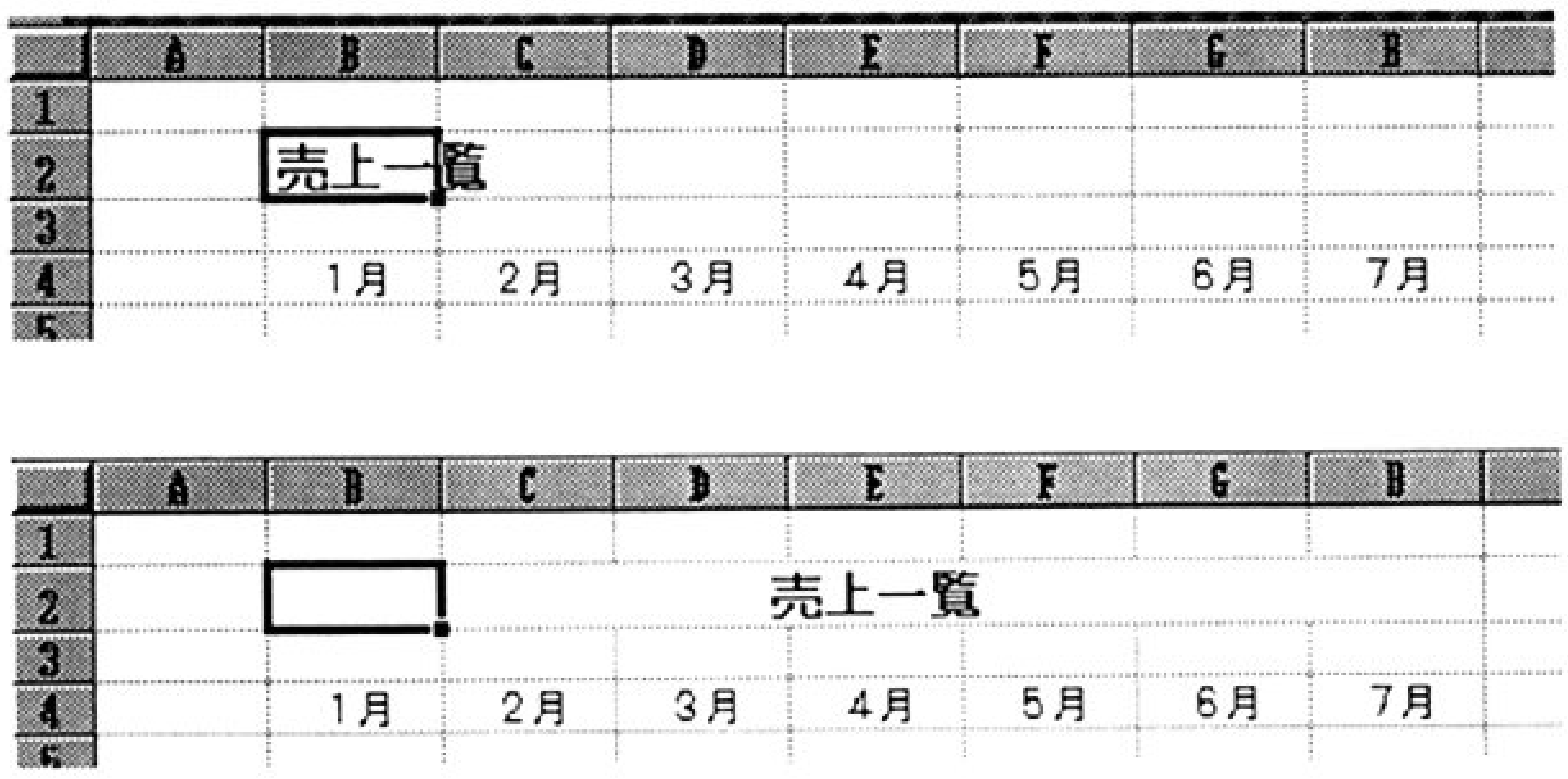
〈図7-3〉セル範囲を選択



〈図7-4〉 セルを確認したいとき



〈図7-5〉 タイトルを記入する



4)．

format 以外の引数は，書式設定をする際に，

number …………表示形式
font ………………フォント
alignment ………配置
border……………罫線
pattern …………パターン
width ……………セル幅/高さ

を指定するかどうかを決めるためのものです．

AutoFormat では，セル幅は列の中のうち最大の長さに設定します．例えば，セル幅を元のままにしておきたければ，

```
Selection.AutoFormat format := xl3DEffects2,
                     width := False
```

のように記述すれば，セル幅は変わりません．これらの引数を省略すると，その項目の設定を行うことになります．

セル内の文字の配置

セル内に，どのように文字を配置するかも重要な問題です．先ほど消した枠線を，もう一度表示してください．そこでまず，セル内の数値（通常右詰め）や文字列（通常左詰め）の横位置を設定するプロパティ **HorizontalAlignment** を紹介します．

このプロパティの値は，組み込み定数（注2）で指定するとよいでしょう．

xlGeneral ……………………標準
xlLeft …………………………左詰め
xlCenter………………………中央揃え
xlRight ………………………右詰め
xlFill …………………………繰り返し
xlJustify ……………………両端揃え
xlCenterAcrossSelection …選択範囲内で中央
xlDistributed ………………均等（漢字圏のみ）

例えば，B1 のセル内の文字をセンター合わせにしたいのならば，

```
Range("B1").HorizontalAlignment = xlCenter
```

と記述すればよいのです．

便利なのは，**xlCenterAcrossSelection** です．表のタイトルなどを適当な位置に入力してあるとします（図7-5）．タイトルは表の中央に来るようにしたいですね．そこで，タイトルの書いてあるセルを含めて選択し，この定数を指定します．

```
Range("B2 : H2").select
Selection.HorizontalAlignment=
                    xlCenterAcrossSelection
```

これで，タイトルを選択範囲の中央（表の中央）に配置することができました（図7-6）．

xlDistributed は，中国，日本などの漢字を使う場合にのみ有効で，半角文字のみを含むセルでは何もしません．

```
Range("B2").select
```

（注1）「デバッグウィンドウを開いていると，何も操作ができないのにお前はどうやっているのか」というお叱りをいただきました．シートの選択などのコマンドを選択する際には，デバッグウィンドウのコントロールボックスをダブルクリックして閉じてください．

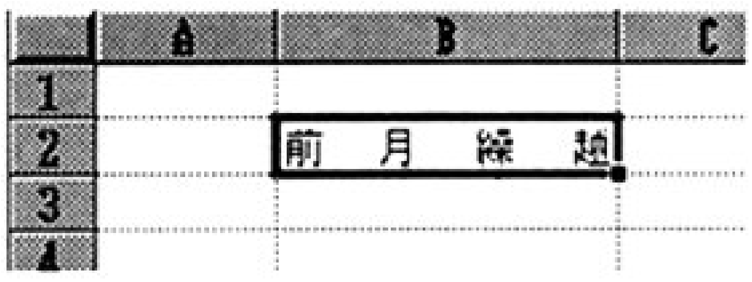
モジュールシートに戻り，「CTRL+G」でデバッグウィンドウを開けば，位置，大きさ，内容は以前のままで使うことができます．

（注2）「組み込み定数の実際の値を教えて」という質問がありました．しかし，値の取得も設定も記号で記述した方が便利ですので，実際の値はあまり意味がないと思います．

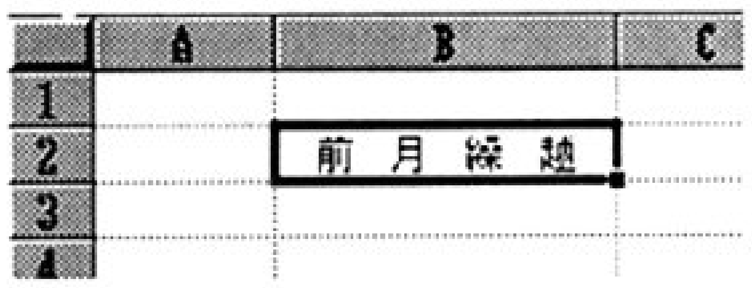
実際の値を利用して，何かトリッキーなコーディングが可能だとお考えなのでしょうが，意味がある場合は少ないと思います．その証拠として，**HorizontalAlignment** の定数の実際の値を示しましょう．

xlGeneral …………………………1
xlLeft ……………………………−4131
xlCenter…………………………−4108
xlRight …………………………−4152
xlFill ………………………………5
xlJustify ………………………−4130
xlCenterAcrossSelection ………7
xlDistributed …………………−4117

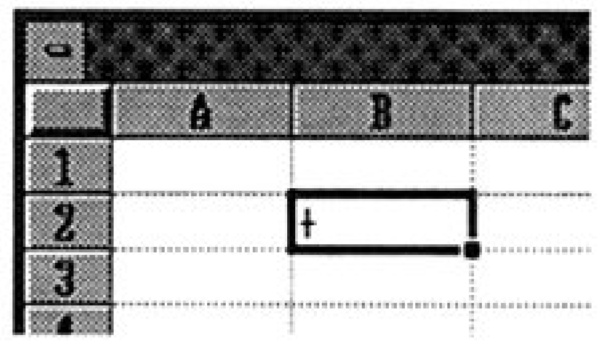
〈図7-7〉字間調節前



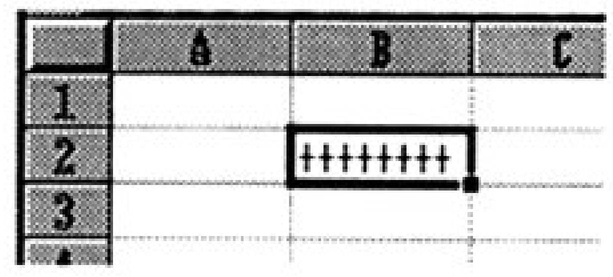
〈図7-8〉字間調節後



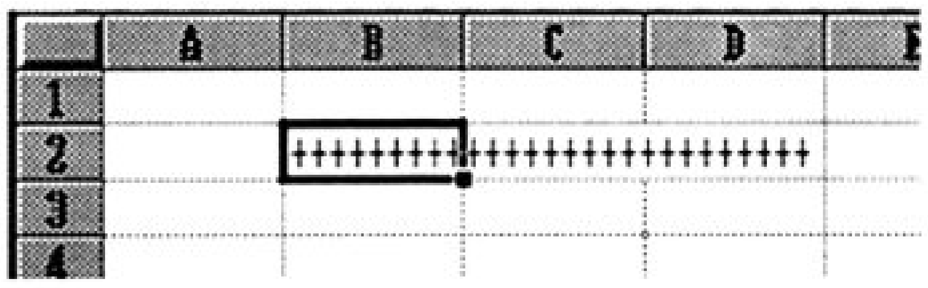
〈図7-9〉コピー前



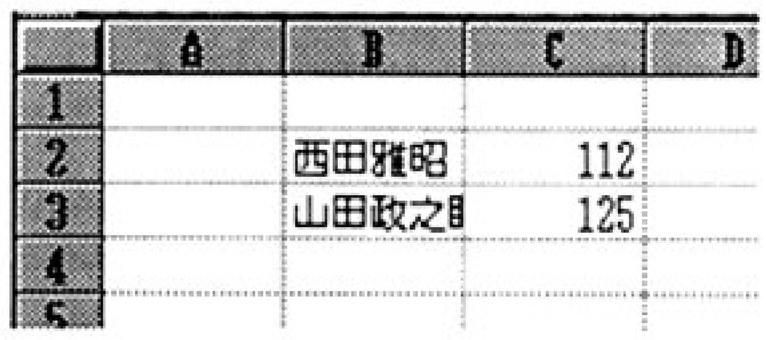
〈図7-10〉コピー後



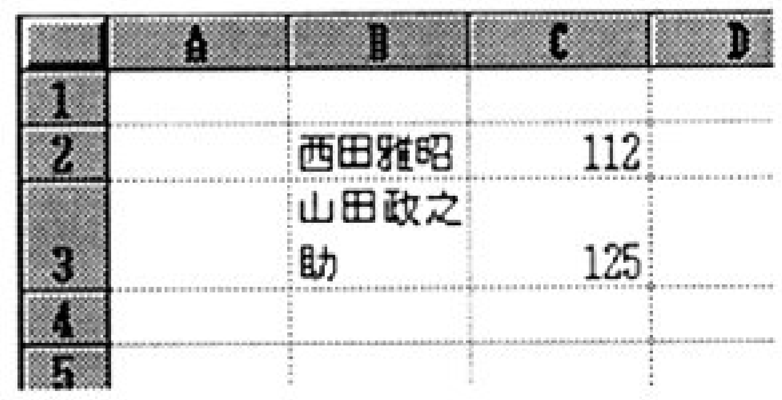
〈図7-11〉空白セルにコピー



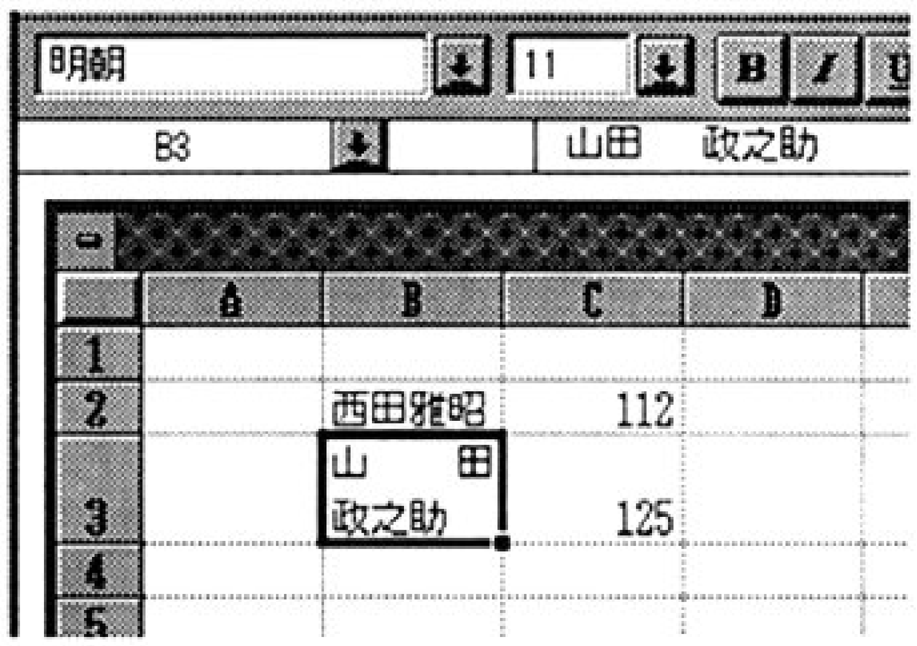
〈図7-12〉入力文字が切れる



〈図7-13〉xlJustify を使うと



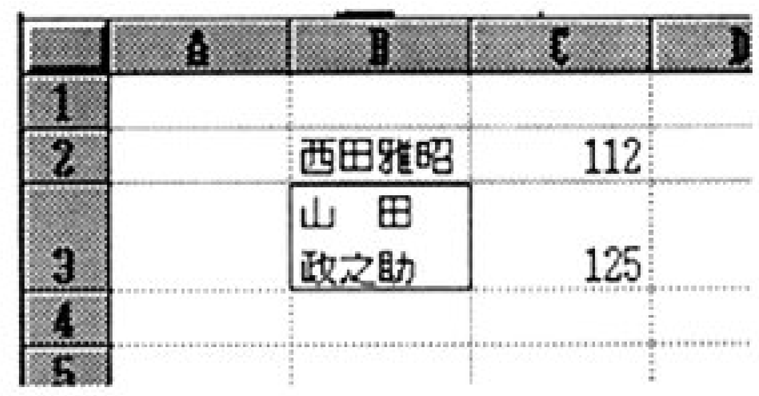
〈図7-14〉スペースを入れる



〈図7-15〉Wraptext を使う



〈図7-16〉テキストボックスを使う



Selection.HorizontalAlignment = xlDistributed
だけでは**図7-7**のようになって，少し体裁がよくないのですが，

Selection.AddIndent = True
という設定を付け加えると，**図7-8**のように美しくなります．**AddIndent**は，文字列の前後にスペースを付けるプロパティなのです．

xlFillは，現在のセルの内容を，いくつもコピーしてセル幅一杯に表示します．例えば，**図7-9**に対して，

Range("B2").Select
Selection.HorizontalAlignment = xlFill
とすると**図7-10**になります．面白いのは，

Range("B2 : D2").Select
Selection.HorizontalAlignment = xlFill
のように，選択したセルの右にいくつかの空白セルがあり，そのセルも含めて**xlFill**の設定を行うと，そのセルまで表示することです(**図7-11**)．飾りなどを付けるときに便利かもしれません．

xlJustifyに関しては，少し説明が必要です．一つのセルの中に複数行の文字列を書きたい場合があります．

図7-12では，B3の文字は後が切れてしまいます．このような場合，

Range("B3").Select
Selection.HorizontalAlignment = xlJustify
と指定すると，**図7-13**のように全体を2行に表示してくれます．しかし，文字列の切れ目が不愉快な上に，何も**justify**をしていないように見えます．

しかし，よく見ると「西田雅昭」と「山田政之」では，文字列の幅が少し違っています．そこで「山田」の後に半角スペースを3つほど入れてみました(**図7-14**)．なるほど1行目だけは**justify**するのですが，これはバグではないでしょうか．

しようがないので，文字列を列幅で折り返すプロパティ**Wraptext**を使ってみました．

Selection.Wraptext = True
これでも，うまくいきません(**図7-15**)．ただし，1行目の文字列には何もしていません．そこでもう一度，

Selection.HorizontalAlignment = xlJustify
をやってみたのですが，**図7-13**と同じ結果になってしまいます．

テキストボックスの利用

こうなると，どうしても美しい文字の配置にしてみたくなります．今回の「セル内の文字の配置」とは関係ありませんが，「テキストボックス」を使うことにしました(**リスト7-1**)．

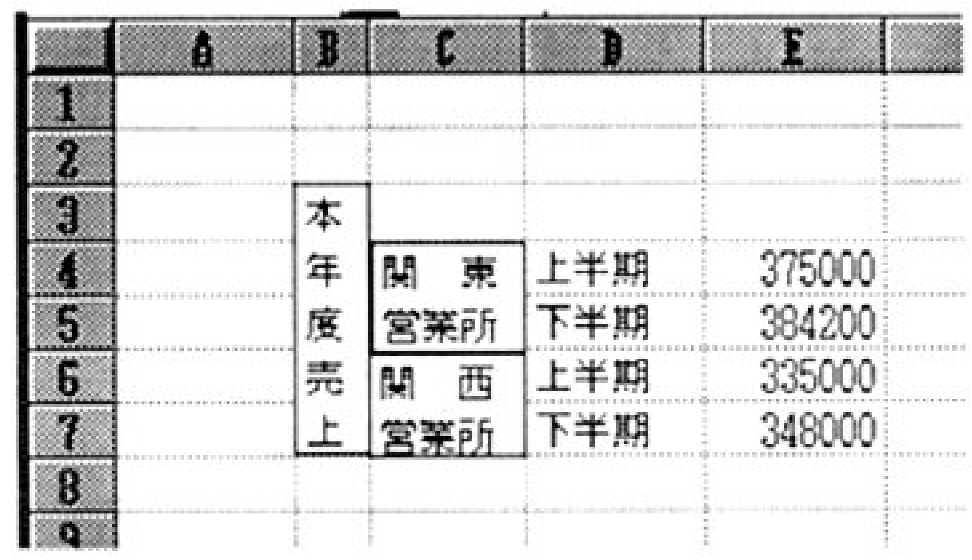
このコードに関しては，また別の機会に説明します．差し当たっては「マクロ記録」で，「テキストボックス」ツールボタンを利用してコードを作ってください．

今度はうまくいきました(**図7-16**)．セルを使用した方法では，行の高さが一緒に変わってしまいますので，表などのラベルには向きません．このテクニックを使えば，**図7-17**のような見やすい表を作ることも可能で

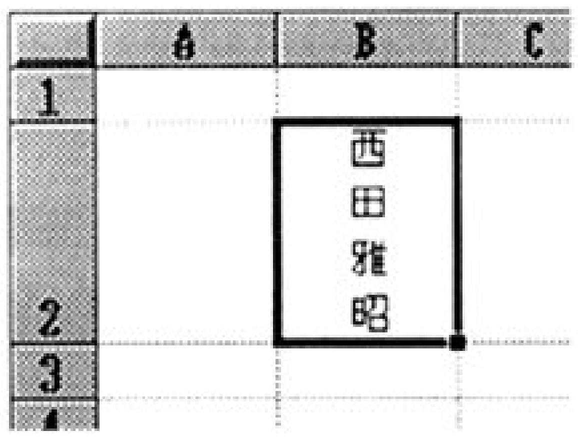
〈リスト7-1〉 テキストボックスを使うプログラム

```
    ActiveSheet.TextBoxes.Add(42, 26.25, 42, 24.75).Select
    Selection.Interior.ColorIndex = 2
    Selection.Characters.Text = "山　田 ↵
政之助"
    With Selection.Characters(Start:=1, Length:=6).Font
        .Name = "明朝"
        .FontStyle = "標準"
        .Size = 11
        .ColorIndex = xlAutomatic
    End With
```

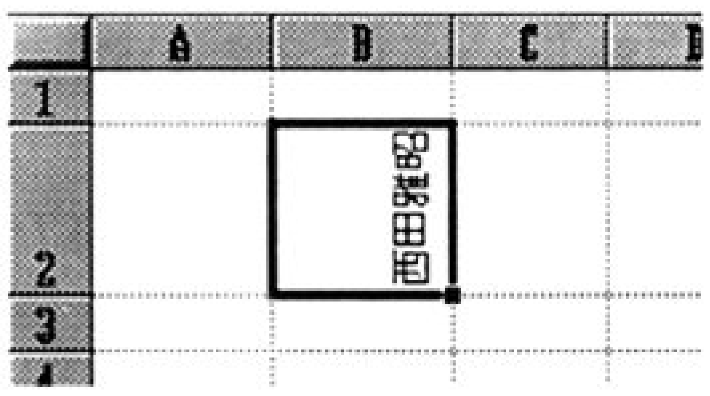
〈図7-17〉 文字表示を見やすくした例



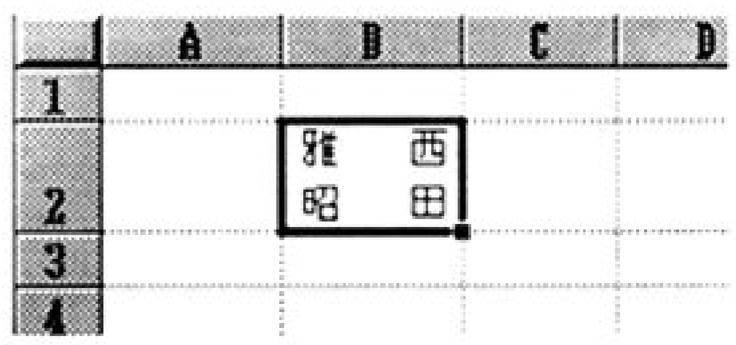
〈図7-18〉 縦表示



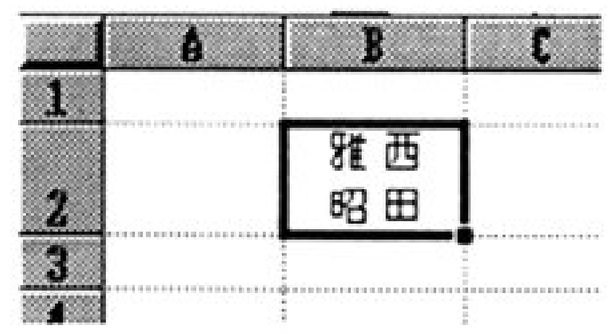
〈図7-19〉 xlUpward を使う



〈図7-20〉 xlVertical を使う



〈図7-21〉 xlGeneral を使う



しょう．

縦位置の文字配置は，**VerticalAlignment** プロパティを使います．ここで使われる組み込み定数は，

- **xlTop** ……………上詰め
- **xlCenter**…………中央揃え
- **xlBottom** ………下詰め
- **xlJustify**…………両端揃え
- **xlDistributed**……均等(漢字圏のみ)

です．これは説明する必要はないでしょう．横書きの場合は，行の高さが大きい場合に問題となるだけですが，縦書きの場合は横書きのときと同じことが言えます．

そこで最後に，文字の縦書き，横書きについて説明しておきます．これには **Orientation** プロパティを使います．このプロパティの組み込み定数は，

- **xlHorizontal** ……横書き
- **xlVertical** ………縦書き
- **xlUpward** ………横書き縦読み，下から上に
- **xlDownward** ……横書き縦読み，上から下に

となっています．

仮に「西田 雅昭」という文字列が B2 セルにあると仮定しましょう．ここで，

Range("B2").Select
Selection.Orientation = xlVertical

とすると，図7-18 のようになります．これは当たり前ですが，

Selection.Orientation = xlUpward

ではどうなるでしょうか．この場合は，文字列が右によってしまいます(図7-19)．同様に，**xlDownward** ならば左によります．このことをよく理解しないと美しい表は作れません．もちろん，

Selection.HorizontalAlignment = xlCenter

で中央に戻すことができます．しかし，列全体が縦書きの場合には，

Columns("B").AutoFit

で，セル幅を自動調整するのがよいでしょう．

面白いのは，横書きの標準の状態から

Selection.HorizontalAlignment = xlJustify
Selection.VerticalAlignment = xlJustify

とやっておいてから，

Selection.Orientation = xlVertical

とすると，図7-20 のようになります．ここで，

Selection.HorizontalAlignment = xlGeneral

で，図7-21 のような書式も作れます．コマンドの順番によって結果が違う場合もありますから，いろいろ実験してみてください．

今回は，文字の配置について少し詳しく説明しました．次回は，色や罫線の説明をする予定です．

VBA プログラミングアドバイス

プログラムの実行　イベントとは

前回，ようやく一本のプログラムができあがりました．今回はプログラムの作成は一休みして，プロシージャの実行について説明します．

VBA のプロシージャは，「イベント」によって起動されます．イベントには，ブックの開閉，ワークシートの再計算，マウスのクリック，キーの操作など，いろいろあります．ユーザーが発生するイベントもあれば，Excel が発生するイベントもあります．

プロシージャの名前によって，どのイベントで実行するかがきまる「自動実行プロシージャ」と，プロシージャ名は自由ですが，メソッドやプロパティによって，イベントと結び付ける「イベントプロシージャ」があります．

自動実行プロシージャ

プロシージャの名前で実行方法を識別するのは，MS-DOS の **autoexec.bat** でおなじみですが，VBA には三通りの自動実行プロシージャの名前があります．

▶ **Auto_Open**

Auto_Open という名前のプロシージャは，ブックを最初に開いたときに，自動的に実行されます．いろいろな初期設定や，起動画面の作成などに使われます．

前回までに作成したプログラムのように，キーボードマクロをたくさん作っておいて，

```
Sub Auto_Open( )
    MacroA
    MacroB
    MacroC
End Sub
```

のようなモジュールを書くだけで，大きなシステムを作ることができるのです．

▶ **Auto_Close**

Auto_Close という名前のプロシージャは，そのプロシージャを持っているブックを閉じる直前に自動的に実行されます．

ユーザー定義のメニューバーやツールバーをデフォルトのものに戻したり，必要なデータを保存したりするのに使われます．プログラム内で環境を変えた際には，必ず元に戻しておかなければなりません．

▶ユーザー定義

特定のワークシートを開いたときや，閉じたときに実行するプロシージャも作れます．このようなプロシージャの名前は，

ワークシート名!Auto_Open
ワークシート名!Auto_Close
ワークシート名!Auto_Activate
ワークシート名!Auto_Deactivate

の四つです．

ただし，このプロシージャは，特別な利点はありませんので使わない方がよいでしょう．詳しく知りたい方は，「VisualBasic ユーザーズガイド」の 362 ページを参照してください．

また，「起動ディレクトリ」(通常，**excel¥xlstart**)に保存されているブックは，Excel の起動時に自動的に読み込まれて Open されます．

イベントプロシージャ

イベントプロシージャとは，特定のイベントが発生したときに実行されるプロシージャです．これまでも，いろいろな場面でプロシージャをどのようにして組み込めばよいかを説明しましたが，ここでまとめておきましょう．

● **メニュー**

メニューからユーザーが選択した際に実行するプロシージャは，よく使われます．メニューにプロシージャを組み込む方法はいろいろあります．

①「新規マクロ記録」ダイアログボックスで「オプション」ボタンを選択して「ツール」メニューに登録
②「マクロ実行」ダイアログボックスで「オプション」ボタンを選択「ツール」メニューに登録
③「メニューエディタ」の「マクロ名/参照」コンボボックスで，作成済みのマクロ名を指定

● **ツールバーボタン**

ツールバーのボタンにプロシージャを登録しておくと，ユーザーは楽にジョブを選択できます．ただし，この場合は，メニューへの登録もしておく方がよいでしょう．登録の方法は二通りあります．

① ボタンをツールバーに登録した際に「マクロの登録」ダイアログボックスに作成済みのマクロを登録
② 既存のボタンにプロシージャを登録

「ツールバー」ダイアログボックスを開いている状態

で，目的のボタンをマウスで右クリックすると，ショートカット・メニューが開くので，「マクロの登録」を選択，そこで作成済みのマクロを登録

● **図形オブジェクトとコントロール**

コマンドボタンやスクロールバーなどにプロシージャを登録しておけば，GUIらしいユーザーインターフェースを作ることができます．

① コマンドボタン作成時

ボタンを作成した瞬間に「マクロの登録」ダイアログボックスが開きます．

② 作成済みのコントロールに登録

コントロールで右ボタンクリックでショートカットメニューを開き，「マクロの登録」を選択して，作成済みのマクロを登録．

「マクロの登録」ダイアログボックスでは，「記録」ボタンを選択すると，マクロの自動記録を行うことができます．また，「新規作成」ボタンをクリックすると，モジュールシートに移って，新たにプロシージャを作成することもできます．

この場合のイベントは，コマンドボタン，チェックボックス，オプションボタンなどでは，ユーザーがクリックした際に発生するので，プロシージャ名は

```
Sub ボタン1_click( )
   ⋮
EndSub
```

のようになります．

エディットボックスやスクロールバーの場合は，ユーザーが何らかの変更を加えた際にイベントを発生するので，プロシージャ名も

```
Sub エディット1_Change( )
   ⋮
End sub
```

のようになります．

プログラムによるイベントの設定

これまでに説明してきた操作によるのではなく，プログラムの上でイベントを設定することもできます．そのために用意されたプロパティやメソッドを次に紹介します．

▶ **OnAction** プロパティ

OnAction プロパティは，オブジェクトがクリックされたときに実行するプロシージャの名前を返します．

例えば，

```
Application.Toolbar("xlWorksheet").Menu("ツール")
.MenuItem("罫線消去").OnAction = "DeleteGrid"
```

と記述すると，ワークシートのメニューバーの「ツール」の「罫線消去」というコマンドを選択すると，**DeleteGrid** というプロシージャを実行するようになります．

筆者は，先に述べた **Auto_Open** プロシージャと **OnAction** プロパティの組み合わせが，最もすっきりしたプログラムの作り方だと思っています．

なお，**OnAction** プロパティの使えるオブジェクトは，図7-Aのとおりです．

表7-A に，**OnKey** メソッドと **OnTime** メソッドの内容と引数を示します．これらによって，実行するプロシージャと時間を指定することができます．

OnKey メソッドの引数 **procedure** に空の文字列 "" を指定すると，Excelが持っているキーストロークの設定を無効にすることができるので，あるキーの機能

〈図7-A〉
OnAction プロパティの使えるオブジェクト

```
Arc, Arcs, Button, Buttons, ChartObject, ChartObjects, CheckBox, CheckBoxes,
DialogFrame, Drawing, DrawingObjects, Drawings, DropDown, DropDowns, EditBox,
EditBoxes, GroupBox, GroupBoxes, GroupObject, GroupObjects, Label, Labels, Line,
Lines, ListBox, ListBoxes, OLEObject, OLEObjects, OptionButton, OptionButtons,
Oval, Ovals, Picture, Pictures, Rectangle, Rectangles, ScrollBar, ScrollBars,
Spinner, Spinners, TextBox, TextBoxes, ToolbarButton
```

〈表7-A〉
OnKey メソッドと OnTime メソッド

● **OnKey メソッド**

m	**Application.OnKey(key, procedure)**	特定のキーをたたいたときに実行するプロシージャを指定する．
引数	**key**	必ず指定する．キーストロークを示す文字列を指定する．
	procedure	プロシージャ名を示す文字列を指定する．

● **OnTime メソッド**

m	**Application.OnTime(earliestTime, procedure, latestTime, schedule)**	
	指定した日付けと時刻に，指定したプロシージャを実行する．	
引数	**earliestTime**	このプロシージャを実行する時刻を指定．
	procedure	実行するプロシージャの名前を指定．
	latestTime	プロシージャを実行できる最終時刻を指定．これらの引数を省略すると，Excelはプロシージャが実行できるまで待つ．
	schedule	**False** を指定すると，**OnTime** プロシージャの設定を解除する．

を無効にしたい場合に便利です.

OnKey イベントプロシージャは, 他のプロシージャが実行中の場合は, 無効になります. ですから, 実行中のプロシージャを中断したり, 実行を終了させるようなルーチンは書けません.

また, **OnTime** メソッドを用いて,

```
Application.OnTime earliestTime :=
TimeValue("12:00:00"),procedure :=
"timeForLunch"
```

と指定しておくと, 12時になると **timeForLunch** という名前のプロシージャを起動することができます.

ただし, 引数 **earliestTime** に指定した時刻に他のプロシージャを実行している際には, **OnTime** メソッドで指定したプロシージャは実行できません.

latestTime で指定した時刻までに Excel が待機モードにならないとき, 指定したプロシージャは実行されません.

▶ **OnSheetActivate** プロパティ

特定のシート, または指定したブック内の任意のシートをアクティブにしたときに実行するプロシージャを指定します. 特定のシートに特定のメニューバーやツールバーを組み込むときなどに便利です.

Application, Workbook, Chart, DialogSheet, Module, Worksheet などでも指定できます.

▶ **OnSheetDeactivate** プロパティ

OnSheetActivate プロパティと同じような使い方ですが, こちらは指定したシートを離れるときに, 指定したプロシージャを実行します. 例えば,

```
Workbooks("財務").Worksheets("総勘定元帳").
        OnSheetActivate = "SpecialMenu"
Workbooks("財務").Worksheets("総勘定元帳").
        OnSheetDeactivate = "UsualMenu"
```

の2行を書いておくと, 「財務」というブックの中で, 「総勘定元帳」というシートに入ると, **SpecialMenu** というプロシージャを実行し, 抜け出るときには **UsualMenu** というプロシージャを実行します.

なお, この二つのプロパティの設定を解除するには, プロシージャ名に空の文字列 "" を次のように指定します.

```
Application.OnsheetActivate = ""
```

▶ **OnWindow** プロパティ

前の二つと似ていますが, ユーザーがジャンプコマンドを選択したり, マウスでウィンドウをクリックしたときに, 指定したプロシージャを実行します. ただし, 滅多に使うことはないでしょう.

▶ **OnCalculate** プロパティ

ワークシートを再計算した直後に実行するプロシージャを指定します. 再計算した後で, 列幅の自動調整などに利用できます.

▶ **OnEntry** プロパティ

ユーザーがワークシートでデータを入力し, Enter キー(リターンキー)を押すか, 別のセルを選択した際に実行するプロシージャを指定します. 入力したデータのチェックなどに用います.

▶ **OnData** プロパティ

Excel 以外のアプリケーションからデータを受け取った際に実行するプロシージャを指定します. DDE や OLE を使用する際に使います.

▶ **OnUndo** メソッド

「編集」→「元に戻す」コマンドで実行するプロシージャを設定します.

▶ **OnRepeat**

「編集」→「繰り返し」コマンドで実行するプロシージャを設定します.

次回からは, 総勘定元帳などの少し大きなプログラムを作ることにします.

ビギナーのための



Windows Programming FAQ 3　連載

リストボックスで テキストを右揃えにする/他

山本　孝

Q. リストボックスでテキストを右揃えにする

リストボックスに数値を右揃えにして表示させたいのですが，どうもうまくいきません．リスト3-1のようにwsprintfを使用して，リストボックスの中に入れる文字列を揃えて入れてみたのですが，図3-1のようになってしまいます．

リスト3-1のようにすることによって，10桁目で右揃えになるはずだったのですが，なぜかバラバラの位置に書かれてしまいます．また，これでうまくいったとしても，リストボックスのウィンドウの右端に揃えられるかどうか不安です．

どうしたらよいのか教えてください．

A. 固定ピッチフォントを指定する

● 固定ピッチフォントを指定

これは，システムフォントが可変ピッチであるためによく起きる現象です．可変ピッチとは，一文字一文字それぞれの幅が違うフォントのことで，例えば「W」と「I」では幅がずいぶん違いますね．

この可変ピッチの問題は，固定ピッチのフォントを指定することで解決できます．固定ピッチのフォントは，特にフォントの形にこだわらなければ，**Fixedsys**という名前の固定ピッチフォントがあるので，これを使用しましょう．

フォントを作成するには**CreateFont**関数などを使用するのが普通ですが，特定の定義済みフォントは**GetStockObject**で獲得できるので，これを使用します．

コントロールで使われるフォントを変更するには，**WM_SETFONT**メッセージをコントロールに送ることで変更できます．

というわけで，**WM_INITDIALOG**の処理中などにリスト3-2の1文を入れれば一発で解決です(図3-2)．

● リストボックスの中身を自分で書く

固定ピッチフォントを使う以外にもう一つ方法があります．

それは，リストボックスの中身を自分で書くという方法です．リストボックスには**LBS_OWNERDRAWFIXED**と**LBS_OWNERDRAWVARIABLE**というオーナー描画のスタイルがあります．

前者はリストボックスの項目の高さがすべて均一になるスタイルで，後者は一つ一つの項目の高さを変えられるスタイルです．

どちらも，リストボックスの項目が描画されるタイミングのときに親ウィンドウに対して**WM_DRAWITEM**メッセージが来るので，このメッセージを処理すれば項目の内容を自分で書くことができるようになります．

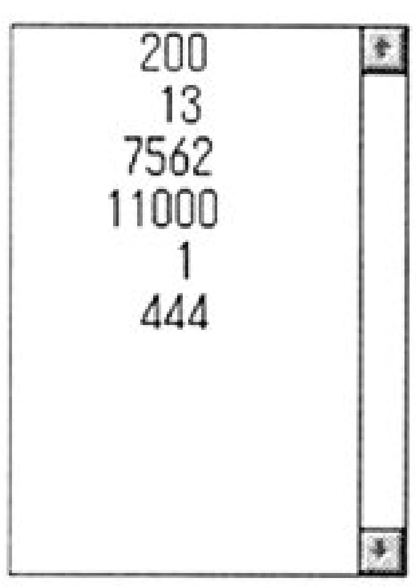
リスト3-3のようにすることで，とりあえず右揃えに表示することが可能になります(これは**LBS_HASSTRING**スタイルも指定した場合)．これで，リストボックスのサイズがどう変わっても，確実に右端に書くことが可能になりました(**図3-3**)．

リスト3-3の**WM_DRAWITEM**処理では，リストボックスがフォーカスを得たときの破線枠の描画や，項目が選択されているときのハイライト表示のことを考えていません．

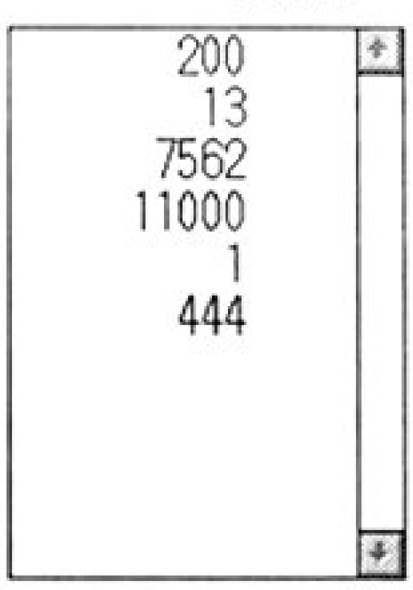
DRAWITEMSTRUCT構造体の中に**itemState**というメンバーがあるので，この状態を見て描画方法を切り分けてやる必要があります．

このほか，余談ですが**WM_MEASUREITEM**メッセージを処理することで，項目の高さを自在に変えるこ

〈図3-1〉リスト3-1を実行すると…



〈図3-2〉右揃えになった

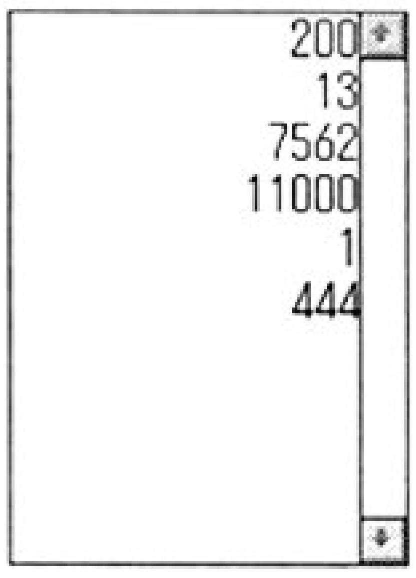


〈リスト3-1〉リストボックスに右揃えで表示させるためにwsprintfを使う

```
BYTE buff[100];
wsprintf( (LPSTR)buff, "%10d", nData );
SendMessage( hList, LB_ADDSTRING, 0, (LPARAM)(LPSTR)buff );
```

〈リスト3-2〉
この1文だけで解決

```
SendMessage( hList, WM_SETFONT, (WPARAM)GetStockObject(SYSTEM_FIXED_FONT), 0 );
```



◀〈図3-3〉
ウィンドウの右端に寄るようになった

〈リスト3-3〉▶
リストボックスの中身を書くプログラム

```
case WM_DRAWITEM:{
  DRAWITEMSTRUCT FAR * dis;
  BYTE buff[100];

  dis = (DRAWITEMSTRUCT FAR *)lParam;
  SendMessage( hList, LB_GETTEXT, dis->itemID, (LPARAM)(LPSTR)buff );
  DrawText(dis->hDC , (LPSTR)buff, -1 ,&dis->rcItem,
           DT_SINGLELINE | DT_VCENTER | DT_RIGHT );

  return(TRUE);
  }
```

とが可能です．

LBS_OWNERDRAWVARIABLE スタイルを指定した場合，項目を追加するたびに Windows からアプリケーションに対して **WM_MEASUREITEM** メッセージを送ってくるので，これに応答してやることで項目ごとに違う高さのリストボックスも作成できるようになります．

図3-4 は，いろいろオーナー描画してみた例です．どれも，とてもリストボックスとは思えない姿ですが，れっきとしたリストボックスなのです．

オーナー描画にすると，少々面倒ではありますが，いろいろなことができるのでバシバシ使っていきましょう．

Q. 縦方向にのみサイズ変更可能なウィンドウ

サイズ変更可能なオーバーラップウィンドウは，必ず縦方向にも横方向にもサイズ変更できますが，これを，縦方向のみにサイズ変更できるようにできないものでしょうか？

とりあえず，WM_SIZE が来たときには MoveWindow とかして調整しているのですが，できればサイズ変更中の枠は横方向には動かないようにしたいんですが．

A. WM_GETMINMAXINFO メッセージに応答する

WM_GETMINMAXINFO メッセージに応答することで，サイズ変更できる最小の大きさと最大の大きさなどを変更することができます．

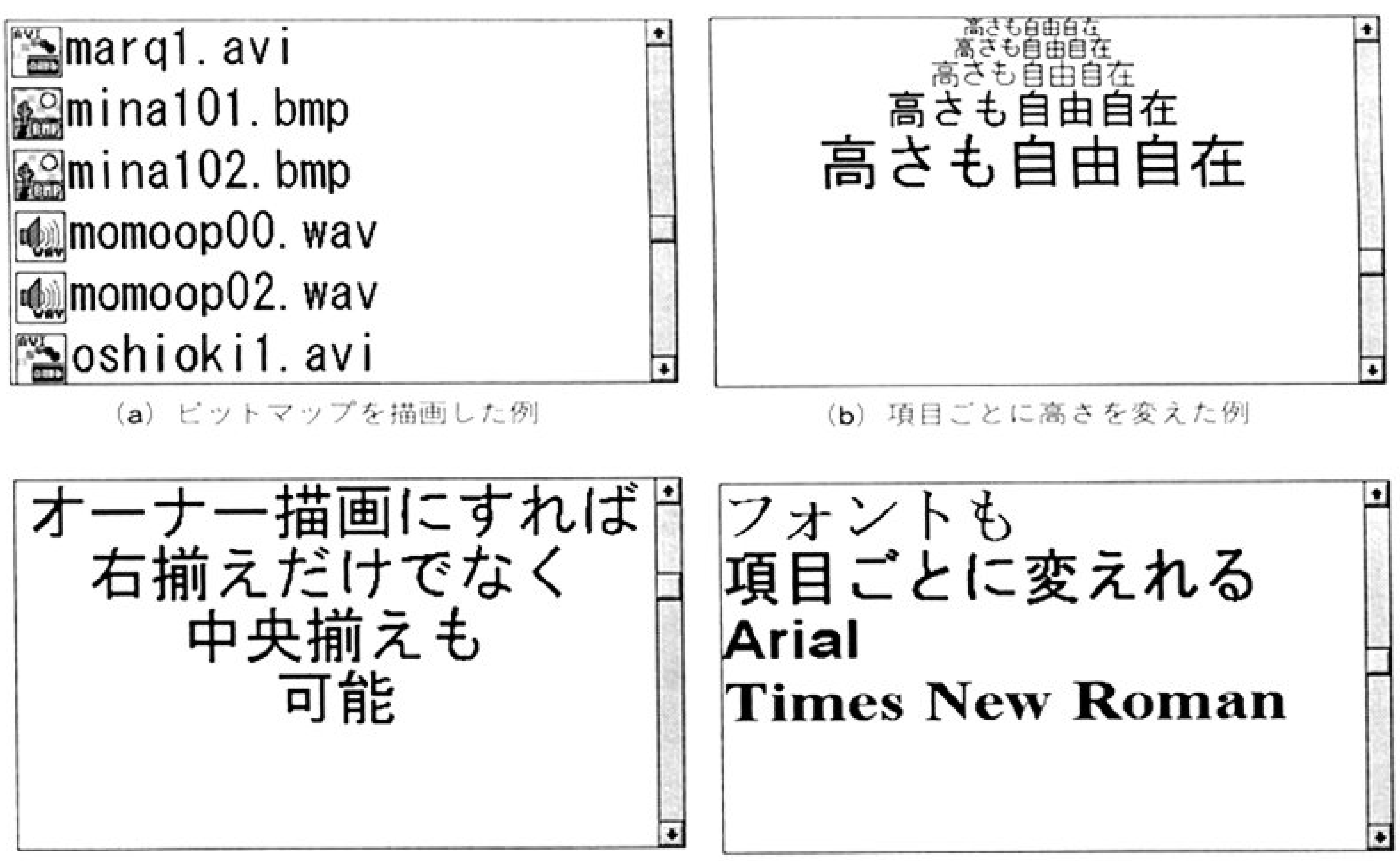
これを利用して，縦方向にのみサイズ変更可能なウィンドウが作成できます．

このメッセージは，SDK のヘルプを見ても，よくわからない説明が書かれているのであまり知られていないようですが，使ってみると**図3-5** のようになります．

WM_GETMINMAXINFO が来たとき **lParam** に **MINMAXINFO** 構造体へのポインタが入っているので，その先の値を変えてやることで，**図3-5** のようなことになります．**図3-5** は，**MINMAXINFO** 構造体の中身をいろいろ変えてみた例です．

この値を，**リスト3-4** のようにすると，縦サイズのみ変更可能なように見えます．なぜそうなるのかというと，**ptMinTrackSize** と **ptMaxTrackSize** の幅を同じにしているので，横方向に動かそうとしても，大きくも

〈図3-4〉
いろいろなオーナー画面



(a) ビットマップを描画した例
(b) 項目ごとに高さを変えた例
(c) 中央揃えで描画した例
(d) フォントを変えた例

〈リスト3-4〉 縦サイズのみ変更可能にする

```
case WM_GETMINMAXINFO :{
  MINMAXINFO FAR * lpmmi;
  lpmmi = (MINMAXINFO FAR *)lParam;
  lpmmi->ptMaxSize.x = 0;          /*最大表示時の位置*/
  lpmmi->ptMaxSize.y = 0;
  lpmmi->ptMaxPosition.x = 640;   /*最大表示時の大きさ*/
  lpmmi->ptMaxPosition.y = 480;
  lpmmi->ptMinTrackSize.x = 300; /*最小のサイズ変更できる大きさ*/
  lpmmi->ptMinTrackSize.y = 100;
  lpmmi->ptMaxTrackSize.x = 300; /*最大のサイズ変更できる大きさ*/
  lpmmi->ptMaxTrackSize.y = 700;
  return(0);
  }
```

できず，小さくもできず，サイズ変更の枠はぜんぜん動かせなくなってしまうからです．

もう一つ考えられる方法として，**WM_NCHITTEST** メッセージを利用する方法も考えられますが，ややこしくなるので **WM_GETMINMAXINFO** を使う方法でいいでしょう．

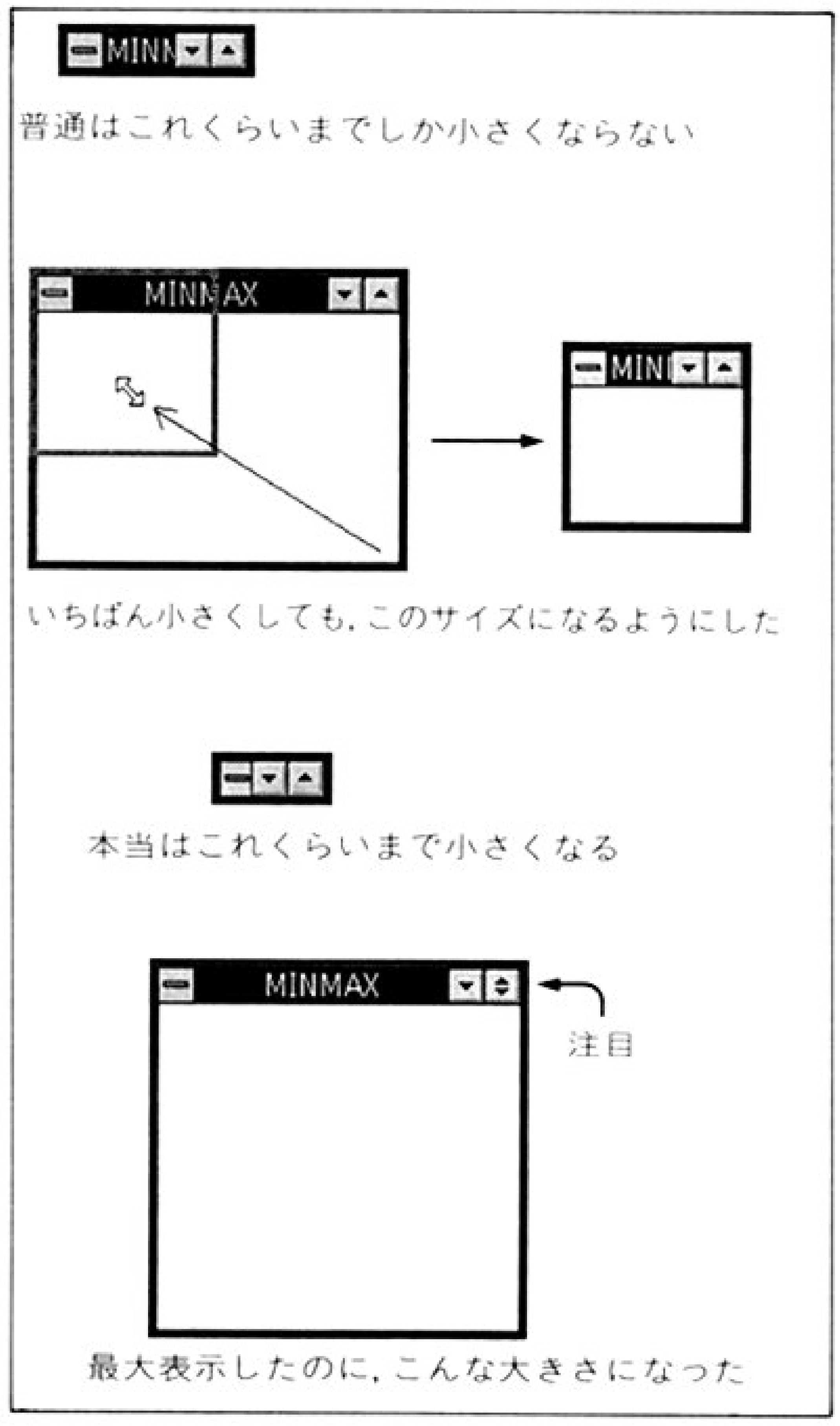
今回は，この二つで質問はおしまいです．

二つじゃ寂しいので，みんなからのお便りドシドシまってるよぉー(ホント)．

＊　＊　＊

このコーナーでは，読者の皆様からの質問を受け付けています．お手紙でも，NIFTY-Serve の ID へのメール(HBG01366)でも何でも OK です．

〈図3-5〉 MINMAXINFO 構造体の中身を変えてみた



パソコンを使った音楽のコンテスト

『第8回デスクトップ・ミュージック「力作」コンテスト』を開催

ローランドは，自作の演奏データや既成曲のアレンジを競う“第8回デスクトップ・ミュージック「力作」コンテスト”を開催する．今回は，同社のデスクトップ・ビデオ・システム「ビデオくん」の発売に合わせて，ビジュアル部門が新設された．

● **応募内容**

オリジナル部門…………自作曲演奏の部門

アレンジ部門……………既成曲の部門

ビジュアル部門(新設)…DTM の演奏にビデオ映像やラジカルムービー，QuickTime，Video for Windows などの映像を組み合わせたマルチメディア作品部門

● **応募方法**

応募用紙と作品の入ったフロッピディスクあるいはビデオテープ(共に5分以内)を事務局に郵送．

● **募集期間**

平成7年6月10日(土)～8月31日(木)

● **賞品**

総合グランプリ
……50万円相当のローランド製品

オリジナル部門グランプリ
……30万円相当のローランド製品

アレンジ部門グランプリ
……30万円相当のローランド製品

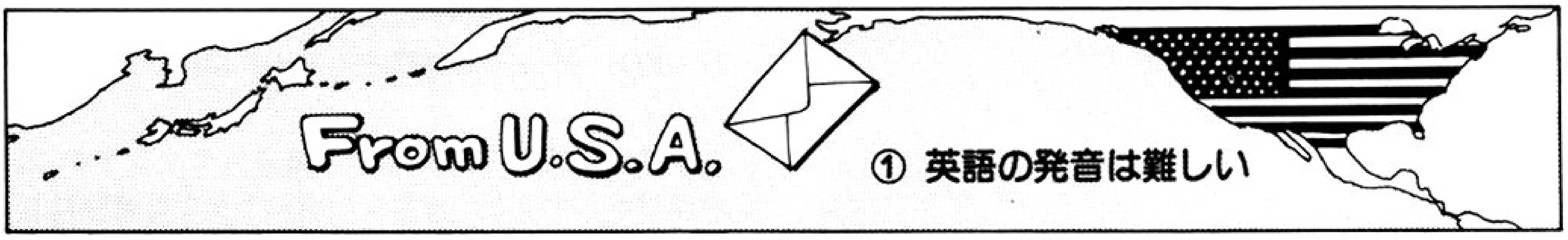
ビジュアル部門グランプリ
……30万円相当のローランド製品

● **データフォーマット**

スタンダード MIDI ファイル．ただし，データ作成ソフトウェアには制限はなし．

その他，詳細については，下記事務局まで．

▶〒101 東京都千代田区神田須田町2-7
ローランド㈱　DTP ホットライン内
デスクトップ・ミュージック「力作」
コンテスト事務局 ☎03-3251-5791



本誌の読者には，コンピュータのプログラムやハードウェアの開発に従事している人，そうでなくても色々なコンピュータについての知識が豊富な人，あるいはそのようになる予定の人など，いろいろいらっしゃると思います．

ここでは，コンピュータ産業の中心地の一つであるシリコンバレーから，最近のトピック，昔の有名なできごとの裏話，あるいは表に出ることなく消えていった話題などを紹介したいと思います．

★ 今回のテーマ

まずは初回を飾る話題として，英語に関する話題，特にカタカナ読みとまったく違う発音(と言い方)のコンピュータ用語と，仕事でよく出てくる(が，日本の英語教育ではあまり使わない)言い回しについてお話します．

コンピュータ用語はほとんど英語ですから，米国に出張しても何とかなると考えている技術者も多いのですが，読み書きはともかく，会話における我々の英語は，米国の子供以下なのでそれなりの覚悟が必要です．

なお，発音記号は使わず，すべてカタカナで発音を表記します．

★ なくなるか r に近くなる t の発音

これはコンピュータ用語に限ったことではありませんが，言葉の最後の 't' の発音がほとんど聞き取れません．

これはまだいいとしても，母音や 'n' の後にくる 't' は 'r' に化けているように聴こえます．代表的なものをあげると，**表1-1** のようになります．

このような単語はカタカナっぽく発音しても九分九厘通じます(というよりはそうしないとだめな場合が多い)．

★ 単語の最後の d や g もほとんど発音しない

“ing”の 'g' もほとんど発音しません．初めて聞くとちょっとまごつく言葉としては，ネットワークでよく使う “Ping” があります．これは「ピン」としか聞こえません．

★ 言い方が違うもの

日本では ROM 焼き機のことを ROM プログラマ(ライター)と言いますが，米国ではそのものズバリ “PROM Burner” です．ROM 抜き機も “PROM Puller” です．

ねじ回しは “Skrew Driver” ですが，プラスのねじ回しは “Cross”，マイナスは “Flat” と言っています．

★「ゥ」に近い！

Beautiful が「ビューリフゥ」になるのと同様に，cable も「ケーブゥ」となります．最悪なのが little でしょう．“Little bit” は「リル ビッ」ですから，“Little Endian CPU” も同様の発音になります．

'r' の発音と聞き間違われるよりは，弱いウにしてしまったほうが相手も間違えません．Apple Computer は「アップゥ コンピューラ」ですし，Intel は「インテゥ」といったほうが通じます．

★ CPU の型番の言い方

英語では西暦と同じように 2 桁ずつまとめていうことが多いようです(奇数桁の場合は，最後の三桁を 1+2 桁にすることが多い)．

ほとんどの CPU の型名はファミリを代表する数字 2 桁+3 桁が多いのですが，これは欧米の数字が 1000 単位で数えられているからです．

★ 日本の英語教育ではなじみのうすい言い回し

次は純粋な英会話の問題なのですが，これが結構多くて困り者ですので，最小限の例をあげておきます(**表1-2**)．

ほかにもいろいろあるのですが，馴染みのない言い回しに出会ったら，素直に聞き返すしかないでしょう．「(日本の英語教育に不備があって)今の意味がわからない」と言ったほうがお互いのためです．相手もそれで言い方を変えてきます．

副詞の pretty なども日本式英語ではなじみがないでしょう．意味は「とっても」という感じなのですが，「プリ」としか聞こえません．日常会話では，How are you doing? と聞かれて本当に調子がよければ，Pretty fine(プリ ファイン)と答えるのが定石となっているようです．なお，副詞の pretty は very なんかよりも使われる頻度が数十倍高いようです．

★ マルティメディア vs マルタイメディア

外来語として一般化している単語のなかで “子音+ィ” と発音する “子音+i” の発音が “子音+アイ” と発音される場合が結構あります．例をいくつかあげると次のようなものがあります．

Semiconductor →セマイコンダクタ
Multimedia →マルタイメディア

〈Macorney E. Saitmayer〉

〈表1-1〉 r に近くなる t

単　語	カタカナ読み	米国人の発音(カタカナで表記)
Inter なんとか	インター xxxx	インナァ xxxx
Center	センター	セナー
Twenty	トゥエンティ	トゥエニィ
Montoray	モントレィ	モノレィ
Water	ウォーター	ウァラァ
Computer	コンピューター	コンピューラ

〈表1-2〉 なじみのうすい言いまわし

How are you doing?	How are you なんていう人は私の周りにはいません
How is your stuff?	今やっている仕事の調子はどう？という感じです
cool!	日本人がよく言う “ナイス” に使い方が似ているようです
How come…	なんで…なの？って感じでしょうか
See ya	You の複数形である ya は結構つかいます

連載

計算機茶話

第18回 コンピュータの使われ方

合阪 省

今回は、最近のニュースからコンピュータに関係してくることについて、思いつくままにお話しすることにします。

銀行が倒産する話

今年になって、イギリスの由緒ある銀行が、デリバティブ取引の失敗のために倒産してしまった、というニュースがありました。

損害は一千億円にも上るということですが、その直接の原因になったのは、ある一人の若手ディーラーの失敗によるものだということです。事実は、果たしてたった一人だけの責任かどうかあやしいところもあるようですが、一人のディーラーに大きな責任を与えていたことは確かなようです。

どうして、そのような状態になるまでチェック機能が働かなかったのかが疑問ですが、結局のところ、銀行の内部のチェック機構そのものに問題があったのでしょう。

それにしても、損害の額の大きさもそうですが、我々の感覚からすると、そんな理由によってあまりにもあっけなく銀行が倒産してしまうという事態に驚かされます。それも、昨日、今日できた中小・弱小の銀行ではなく、数百年の歴史を誇る、英国王室御用達の銀行なのですから。

ちょうど日本では、東京都の二つの信用組合がずさんな経営のためにたちゆかなくなった、という問題で揺れているところに飛び込んできたニュースだったので、なおさらその対比が際立ちました。

相場の世界の高等数学

デリバティブ取引について筆者は全くの素人なのですが、わからないなりに興味をそそられた部分がありました。

それは、この取引にはきちんとした裏付けのある高等数学による理論があって、コンピュータを駆使することによって将来の相場の予測が行われているということです。この理論が作られた背景には、アメリカのNASAを退職した技術者が、金融関係の世界に再就職して開発したことが、そもそものきっかけになったということです。理系の知識を文系の世界に持ち込んだわけです。

ただし、一つの銀行があっという間に倒産してしまったことからもわかるとおり、この取引はいわゆる「ハイリスク・ハイリターン」の取引で、当たれば儲けは大きいのですが、失敗すれば今回のような最悪の結果を引き起こすものです。

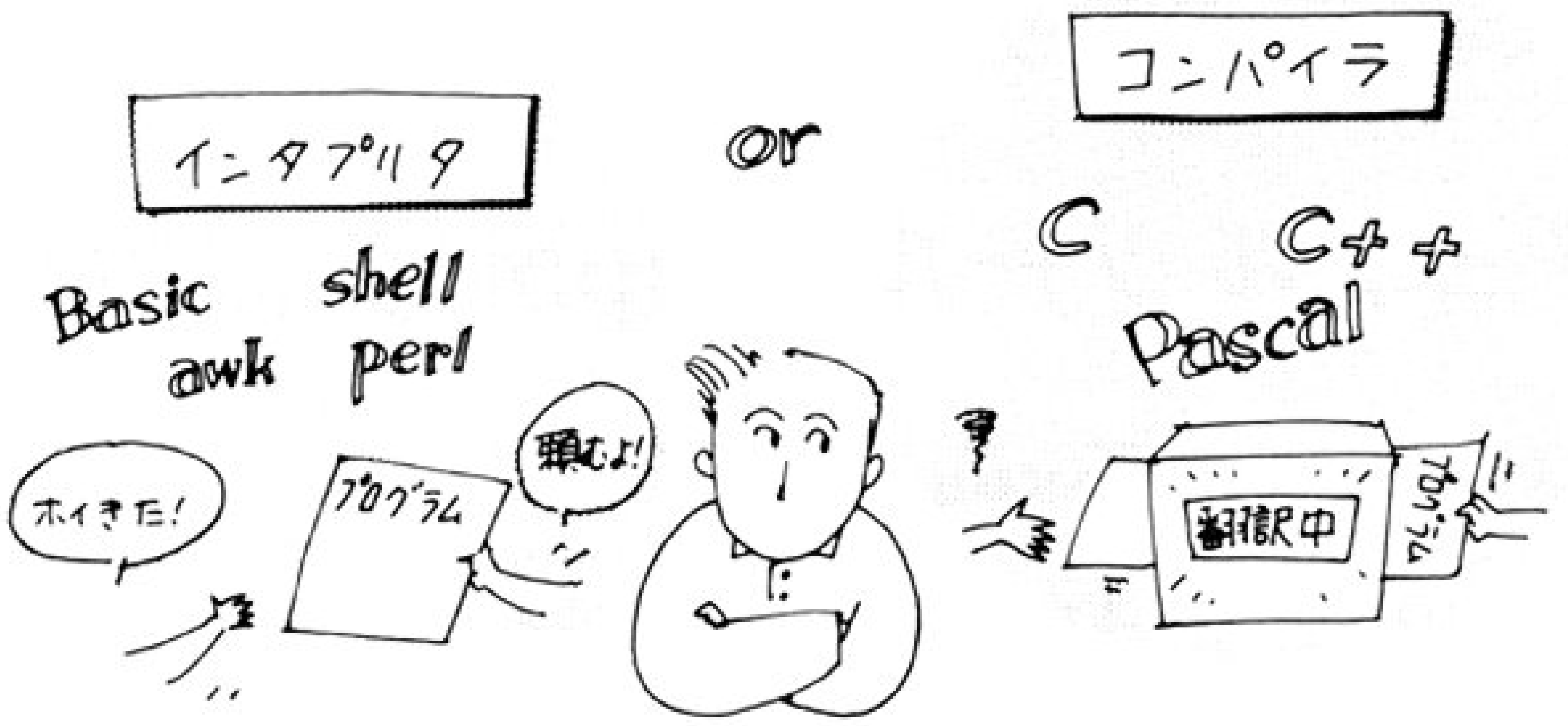
そういうわけで、一つの商品としてこのような取引をユーザーに提供することについては、銀行としてはできれば慎重な姿勢を取っているらしいのですが、同業他社が新商品として提供を始めてしまうと、対抗上やむをえず手を染めなければならないような状況になっているようです。

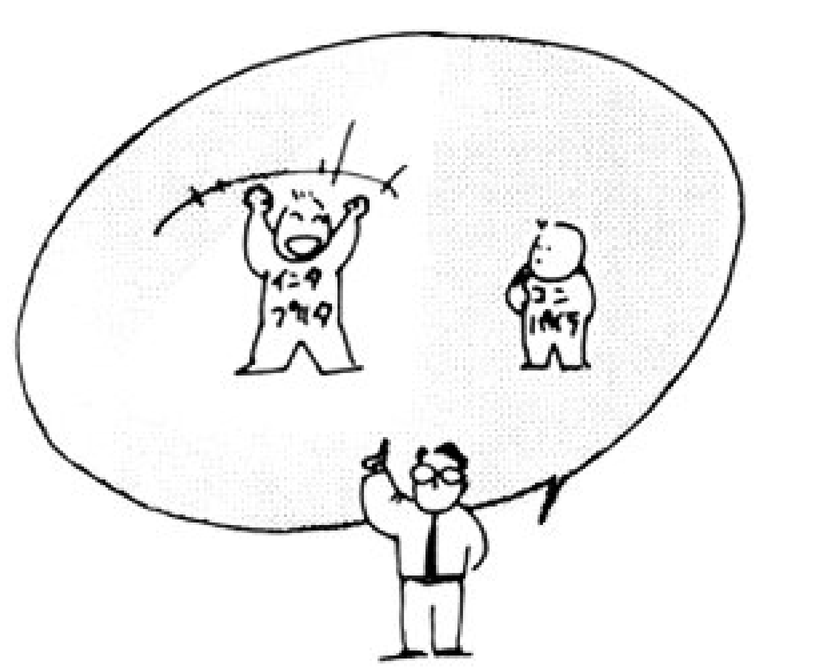
テレビの報道では岡山のある信用金庫が取材されていて、こんな地方の金融機関でもデリバティブ取引に手を染めなければ、これからは生き残っていけないと伝えられていました。

この信用金庫の悩みは、デリバティブを担当できる人を雇うためには、高額の年俸を払わなければならないということでした。デリバティブを担当している役員と同額の年俸を応募者から要求されて、トップが決断を迫られているようすが放送されていました。

そして、長々と書いてきて本題はここだけなのですが、このニュースを見ていたとき、急にUNIXワーク







ステーションがクローズアップされました．するとその上で **vi** が動いていて，何やらＣのプログラムが編集されていて，**ls** の出力で **.o** というサフィックスがついたオブジェクトモジュールが目に入り，妙に印象に残りました．

金融の世界でも，オンラインシステム以外に，こうしてワークステーションが使われているということは以前から話は知っていましたが，それを改めて確認させられた出来事でした．

金融商品の開発にも UNIX＋C による環境が使われているのを見て，無縁だと思っていたディーラーという職業がたいへん身近に感じられたわけです．

コンピュータ言語

というわけで，久々にプログラミングについて考えさせられたのでした．筆者はいろいろな事情から，しばらくプログラミングに携わっていなかったのですが，**vi**＋C の環境を見せられて，この文章を書くきっかけになりました．

読者は，普段どんなプログラミング環境で毎日を過ごしているのでしょうか．本誌の読者層からすると，MS-DOS 上でＣかＣ＋＋，あるいは Windows 上で Visual C＋＋といった方が多いのではないかと思いますが，中には Pascal や MASM の環境を利用している方もいるでしょう．

ここにあげたソフトはすべてコンパイラですが，インタプリタを愛用している方はいないでしょうか．筆者自身，今までの経験の中でずいぶんとＣ言語のお世話になってきましたが，それ以上に UNIX 上の shell，awk，perl といったインタプリタ型の言語を愛用してきました．

awk や perl などは MS-DOS 上にも移植されているので，本誌の読者の中にも愛用している方も多いと思います．文字列を簡単に処理したいときには，これらの言語は本領を発揮します．

コンパイラとインタプリタ

読者の中には，コンパイラとインタプリタの違いをよく理解して，必要に応じて使い分けている方もおられると思いますが，ここでもう一度両者の違いについて考えてみたいと思います．今まで，あまり意識したことがなかったという方は，是非一緒に考えてみてください．

一昔前は，「おなじみの」という風にいえた BASIC 言語も，今となってはなじみのない人の方がはるかに多いのではないかと思います．

しかし，この BASIC こそがインタプリタ言語の代表です．なんといっても，プログラミングとそのプログラムを実行するというサイクルを，気軽に繰り返せるのが大きな魅力でした．気軽にとはつまり，コンパイルすることなしに即実行できるということです．

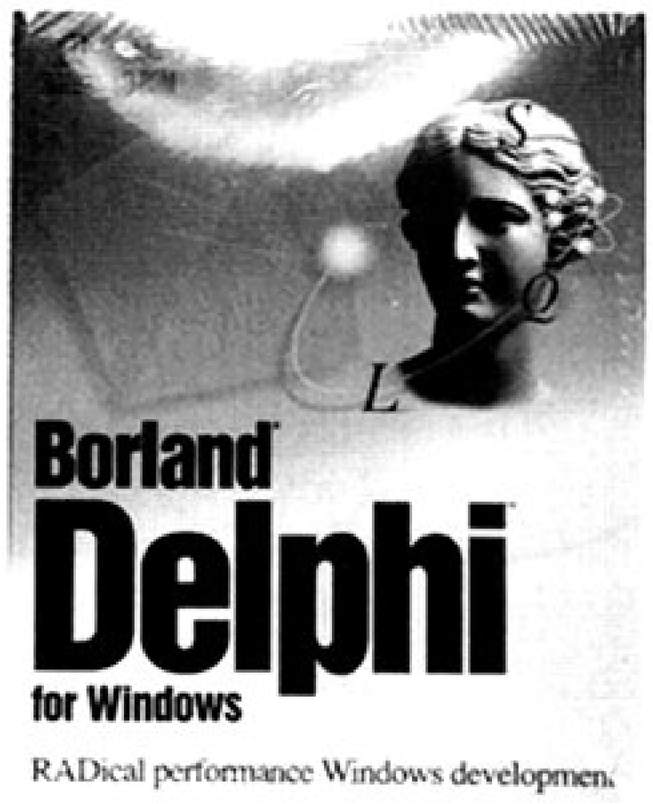
エディタでプログラムを作成して，すぐに実行できることは大きな魅力です．とはいっても，当時のパソコンの性能を考えると，ステートメントを逐次実行していくインタプリタ形式の BASIC はその簡易な構文も裏目にでて，ちょっと複雑なことをさせるにはあまりにも不十分なものでした．

当時は，パソコン上で動くＣのコンパイラのような処理系は，プログラマのあこがれの的だったものです．コンパイラそのものの価格も，個人で購入するには高価なものでしたし，それを満足に動かすためにはメモリの増設も必要で，それがまた高価でした．

現在では，Ｃコンパイラは誰にでも買えるものになり，当たり前の環境になってしまいました．また，ハードウェアの進歩も，たいへんなスピードで進んでいます．

その性質上，インタプリタはプロトタイピングに大変適しているので，ハードウェアの進歩はインタプリタ形式の言語に再び光を当てることになるのではないかと筆者は考えています．

〈㈱VSL　あいさか さとし〉



アプリケーションも開発できる
データベース・アクセスツール

文/船江明彦

ビジュアルツールDelphi(英語版)

Delphi は Visual Basic のような手軽な開発環境に,本格的なプログラム作成機能を備えた開発言語です. 基本的に Visual Basic などと同じく, フォームにコンポーネントと呼ぶツール類を配置し,プロパティ/イベントを組み込んで目的のプログラムを作成するビジュアルツールですが, れっきとしたコンパイラです.

Delphi は Pascal 系言語です. Pascal には馴染みのない方が多いと思いますが, 少なくとも C++の文法のややこしさに比べると覚えることも少なくて簡単です(図1).

また, ビジュアルツールですが, Visual Basic とは異なり, コンパイラによる高速なコード生成が期待できます. さらに, インラインアセンブラの使用, DLL の作成, C++などと同様の継承の使用, Delphi 自体でコンポーネントを作成できるなど, 数多くの特徴があります.

その他, dBASE や Paradox などのデーターベースからクライアント/サーバー型のデータベースまで取り扱うことが可能で, 本格的なシステムの作成にも使用できそうです.

● コンパイラの威力

Visual Basic とは異なり, ネイティブな機械語を出力するので, 実行速度が高速です.

評価用に Delphi で作成した1から10,000までの合計値を求めるプログラムは, 実行時間は約3.1 ms でした(**リスト1**)が, 同じく Visual Basic では(**リスト2**)41 ms で約10倍の開きがあります.

コンパイラと中間言語インタプリタを比較するので, 極めて乱暴な比較ですが, 違いは理解していただけると思います.

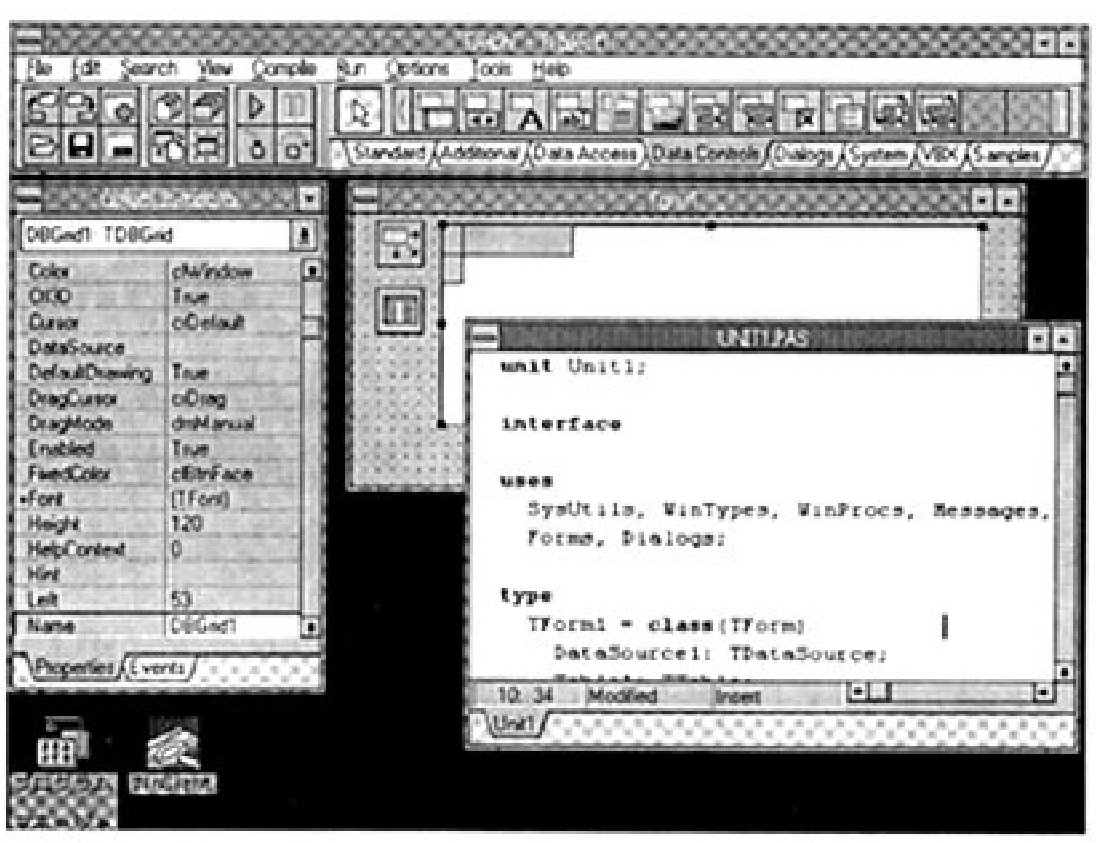
測定に使用したパソコンは, 486DX2-66に交換した PC9801BX2 で, プログラムはボタンのクリックのイベントプロシージャに組み, 時間の測定のためにループの外に大ループを設けました.

● 究極のプログラムレス言語でデータベースファイルのビューワを作成する

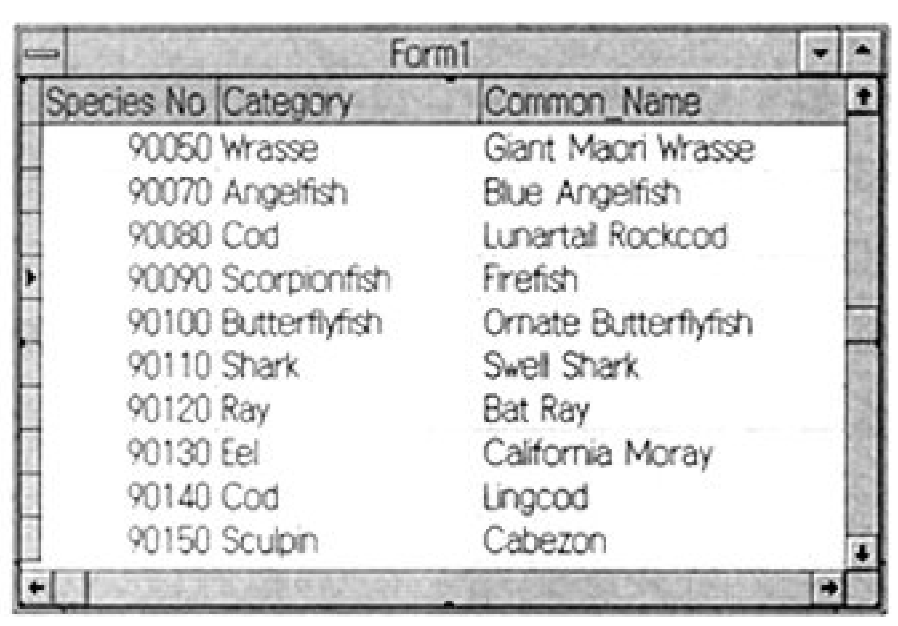
Delphi のコンポーネントを組み合わせることにより, 1行もコーディングすることなく, プログラムを作成することもできます.

図2は, Delphi で作成したデータの一覧表示ができるデータベースのビューワです. dBASE, Paradox などのデータベースの表示, レコード, 項目のスクロール表示などができます.

データベース用のコンポーネントである **TTable**,



◀〈図1〉
Delphi の動作中の画面



▶〈図2〉
Delphi で作成した
データベースビューワ

〈リスト1〉 Delphi で作成した1から10000までの合計値を求めるプログラム

```
procedure TForm1.Button1Click(Sender: TObject);
  var s     : String[20]  ;(文字列型変数 s 定義)
  var i     : integer     ;(16ビット整数  i 定義)
  var a     : longint     ;(32ビット整数  a 定義)
  begin
    a := 0;
    for i:=1 to 10000 do begin
      a := a + i;
    end;
    Str(a,s);                         (合計を文字型に)
    Canvas.TextOut (40,40,s);              (画面表示)
  end;
end.
```

〈リスト2〉 Visual Basic で作成した1から10000までを求めるプログラム

```
Sub Command1_Click ()
  a& = 0                                    '4バイト整数
  For i% = 1 To 10000                       '2バイト整数
    a& = a& + i%
  Next i%
  Debug.Print a&                         '合計を画面表示
End Sub
```

〈リスト 3〉 DLL のサンプルプログラム

```
library InpOutp;                              {DLL指定}
{ポート入力 関数 }
function inportb(port: Word): Word; export;
var w:word;
begin                          {値を返す関数はfunction}
  asm                  {asm文でｲﾝﾗｲﾝｱｾﾝﾌﾞﾗを記述できる}
    mov   dx,port       ;
    in    al,dx         ;
    mov   ah,0          ;
    mov   w,ax          ;
  end;
  inportb := w          ;
end;
{ポート出力手続き }
procedure outportb(port:Word; val:Word);export;
begin {値を返さない関数は手続きと呼びprocedure}
  asm
    mov   ax,val        ;
    mov   dx,port       ;
    out   dx,al         ;
  end;
end;
{モジュール内関数番号定義}
exports
  inportb  index  1,
  outportb index  2;
begin
end.
```

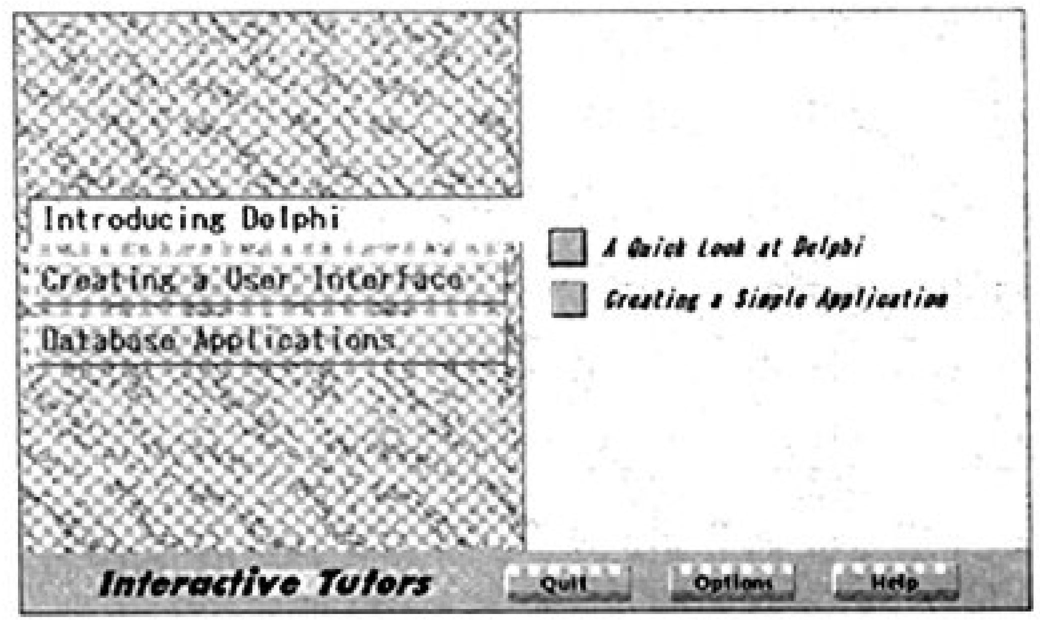
〈リスト 4〉 DLL を Visual Basic から呼び出す例

```
Declare Function inportb Lib "c:¥INPOUTP.dll"(ByVal Port As Integer) As Integer
Declare Sub outportb Lib "c:¥INPOUTP.dll"(ByVal Port As Integer, ByVal value As
Integer)
Sub Command1_Click ()               'ﾎﾞﾀﾝ ｸﾘｯｸ 処理
  If cnt = 0 Then                 '最初にﾎﾞﾀﾝ押した時
    outportb &H37, &H6                  'ﾌﾞｻﾞｰ鳴る
    cnt = 1  'GLOBAL.BASで定義,ﾌｫｰﾑﾛｰﾄﾞで初期化
  Else                              '次にﾎﾞﾀﾝ押した時
    outportb &H37, &H7                  'ﾌﾞｻﾞｰ停止
    cnt = 0
  End If
End Sub
```

〈リスト 5〉 ボタンクリック処理に組み込んだ API 呼び出し

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  ExitWindows(0,0);        {Windowsを終了する。}
end;
```

〈図 3〉 Interactive Tutors



TDataSource, TDBGrid をフォーム上に配置するだけで，このようなビューワを作成できます．

実行ファイルは **EXE** ファイルだけで，別ファイルの DLL などのランタイムを使用することなく動作するので，手軽に配布などもできます．

また，ボタンなどを配置して，処理をつけ加えると検索/更新などの本格的な処理もできます．

● **DLL の作成**

Delphi では，簡単に DLL を作成できます．**リスト 3** は DLL の作成例で，I/O ポートから 1 バイトのデータの入力を行う **inportb** と，出力を行う **outportb** を，インラインアセンブラを使用して定義しています．

また，**リスト 4** は Visual Basic からの呼び出す例です．PC98 用で，ボタンを押せばブザーが鳴り，次に押せば停止します．プログラムは，ボタンのクリック処理に配置し，ブザー鳴動ポートを on/off します．

● **Windows API の呼び出し**

Windows API の呼び出しは DLL の呼び出しと同じですが，Delphi では API の型が定義済みなので，そのまま呼び出せます．

リスト 5 は，Windows システム API の呼び出し例です．Windows API である **ExitWindows** は，そのまま記述できます．ボタンを押すと **ExitWindows** が呼び出され，Windows を終了して DOS に戻ります．

● **GUI が不要のあなたに**

ちょっとした計算プログラムなどでは，マウスやダイアログの GUI は必要ない場合があります．

BASIC の **INPUT, PRINT**，C 言語の **scanf, printf** 程度で十分と思う人には，DOS 風のコンソール入力/出力を代行する機能があります．EasyWin, QuickWin などと同じ機能です．

● **現在発売中の Delphi は英語版**

残念ながら，日本語版は秋頃に発売されるということです．しかし，Interactive Tutors を利用して画面の指示どおりに入力することで，動作やプログラミングを学べます．

前述のデータベースファイルのビューワも，この Interactive Tutors の指示どおりに入力すると作成できます(**図 3**)．

● **データベース系言語としての使用**

マニュアルやサンプルプログラムには，データベースアクセスについて豊富に説明があり，dBASE, Paradox, サーバー系データベースなどにアクセスできます．

したがって，別売の VBX などを使用しなくても，データベースファイルのビューワなどの機能が使用でき，コンパイラ出力で高速な画面などを作成できそうです．

実のところ Delphi は，本格的なビジネス用アプリケーションの作成ツールであるのかもしれません．

帳票画面，伝票入力画面など，細かい部分まで小回りがききそうで，アプリケーション開発用言語としての位置付けが大きいとも思えます．

製品名／価格　Delphi(英語版)／68,000 円
　　　　　　　Delphi Client/Server(英語版)／248,000 円
問い合わせ先　ボーランド(株)
☎ 03-5350-9380



DOSのバージョンアップはつづく

文/山下直彦

IBM PC-DOS Ver 7.0(英語版)の特徴

OS/2やWindows95などによって，DOSが不要な環境に移行しつつありますが，DOSのバージョンアップも続けられるようです．米国で英語版PC-DOS Ver 7.0が発表されましたので紹介します．REXXがサポートされるなどIBM色が強化されています．

今年の3月に，米国IBMより発売されたPC-DOS Ver 7.0について，その特徴を紹介しましょう．

● セットアップ・ディスクのフォーマットXDF

PC-DOS Ver 7.0にはバージョンアップ版もあるようですが，購入のための必要条件が不明でしたので，フルセット版を購入しました．

今回のDOSは，システムディスケットが3.5インチ2HDで計5枚です．そのうちの1枚目を除き，eXtended Density Format(XDF)と称する新しい，特殊なフォーマット形式で供給されています．そのフォーマットは，1.44Mバイトに対し，約1.84Mバイトに容量アップされています．

通常の状態で**DIR**および**CHKDSK**を行うと，図1(a)に示すような表示となります．ファイルとしては**README.XDF**のみが見えます．これは，単純なテキストファイルで，XDFフォーマットディスケットの使用上の注意が書かれています．

また，**XDF.COM**というTSRプログラムがあり，これを実行した後には図1(b)に示すような表示が現れます．エクステンションなしの大きなファイルがいくつか含まれますが，これらはBundle Filesと呼ばれ，いくつかのファイルをまとめたものとなっています．

セットアップの初期に**XDF.COM**が常駐し，インストールの指定に応じて，2枚目以降のXDFフォーマットから，必要なファイルを解凍しながらインストールが行われます．

マニュアルによると，フルインストールをするためにはハードディスクの空き容量が18.5Mバイト必要とありますので，18.5÷(1.44+1.84×4)=2.1となり，特殊フォーマット[注1]の上に，2倍強の圧縮がかかっている勘定になります．

XDF.COMのスイッチとしては，

なし：ロード
/?　：ヘルプ
/u　：アンロード
/h　：ロード(非公開，注2参照)

があります．

XDF.COMは，ディスクBIOSの**INT 13h**をフックして修飾を加えています．

また，個々のファイルの解凍用に**UNPACK2.EXE**というコマンドがあります．**SETUP**ディスケットのNo.1の**FILES.TXT**にファイル一覧がありますので利用できます．

XDFフォーマットのディスケットのバックアップコピーには，**XDFCOPY.EXE**というコマンドが使用で

```
C:\>dir a:

 Volume in drive A is DISK       2
 Directory of A:\

README   XDF       1,373 94-11-17   13:00
        1 file(s)        1,373 bytes
                             0 bytes free

C:\>chkdsk a:

Volume DISK       2 created 1980-00-00 0:00
Errors found, F parameter not specified
Corrections will not be written to disk


   1 lost allocation units found in 1 chains.
         512 bytes disk space would be freed

        2,048 bytes total disk space
        1,536 bytes in 1 user files
            0 bytes available on disk

          512 bytes in each allocation unit
            4 total allocation units on disk
            0 available allocation units on disk

      655,360 total bytes memory
      558,864 bytes free
```

◀〈図1(a)〉フロッピディスクの中を見る

```
C:\>xdf

XDF -- eXtended Density Format
Copyright (c) 1992, 1993 Backup Technologies, Inc.
Copyright (c) 1994, Ametron, Inc.
Patent(s) Pending

C:\>dir a:

 Volume in drive A is DISK       2
 Directory of A:\

DOS1              522,033 94-11-17   13:00
DOS2              479,333 94-11-17   13:00
SHELL2            190,695 94-11-17   13:00
PEN1               84,299 94-11-17   13:00
WUNDEL             99,615 94-11-17   13:00
BACKUP1           409,193 94-11-17   13:00
        6 file(s)      1,785,168 bytes
                          75,776 bytes free

C:\>chkdsk a:

Volume DISK       2 created 1994-12-14 12:20

    1,865,216 bytes total disk space
    1,786,880 bytes in 6 user files
        2,560 bytes in bad sectors
       75,776 bytes available on disk

          512 bytes in each allocation unit
        3,643 total allocation units on disk
          148 available allocation units on disk

      655,360 total bytes memory
      543,152 bytes free
```

〈図1(b)〉▶ XDFによって内部が見える

〈図2〉XDF フォーマットのバックアップ

```
C:\EX>XDFCOPY A: A:
Insert the source diskette in drive A:
Press any key to continue

Reading...
  0%_______25%_______50%_______75%_______100%
  ▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯


Insert the target diskette in drive A:
Press any key to continue

Writing...
  0%_______25%_______50%_______75%_______100%
  ▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯
Verifying...
  0%_______25%_______50%_______75%_______100%
  ▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯
```

きます．2HD フォーマットのディスケットに複写を行うと，XDF 形式でフォーマットしながら記録されます．その実施状況を**図2**に示します．

実行中，ドライブCに一時ファイルを作成して，一度に読み込まれます．

● REXX のサポート

インストール時の選択項目(デフォルトではNO)にREXX が入っています．

標準的な REXX のコマンド，関数などは完全に揃っています．日本語版 OS/2 用と比較すると，IBM の SAA 規約に沿った内容になっています．REXX の V 4.0 レベルに対応しているようです．コマンドにおいて，OPTIONS(DBCS 関連)がない，DBCS 文字列処理関数がないといった程度です．

ただし，REXX ユーティリティ関数がやや少ないという印象をもちましたが，このままでもかなりのプログラムが作成できるでしょう．

デフォルトの拡張子は，バッチファイルと同様の **.BAT** です．1行目の最初が注釈文の **/*** で始まるとき，REXX ファイルと解釈されます．

REXX ファイルの実行は，

(1) **REXX filename**

(2) **filename**

の二つの方法があります．(1)の場合，ファイル名の拡張子は任意で，**REXX.EXE** を直接呼ぶ形となります．

(2)の場合は，拡張子は省略できますが，デフォルトの **.BAT** であるものとされ，**COMMAND.COM** が対応するファイルの初めの2文字を調べ，REXX のファイルであれば，**REXX.EXE** にファイル名を渡して実行を引き継ぎます．実際，**COMMAND.COM** 内に，このための **REXX.EXE** の文字列を見出すことができます．

また，OS/2 の **REXXUTIL** 関数は，DLL の形で供給されますが，DOS の場合は組み込み関数と同様に扱えます．また，トークン化やファイルの保存機能はないようです．

なお，後ほど述べる，PC-DOS 6.3/V との併用により，日本語文字列の **Say** コマンドによる表示なども可能です．ただし，注意点として，半角文字の引用符でくくって使用する必要があります(その内部では，全角の引用符も使用できる)．

● ヘルプファイルの充実

PC-DOS Ver 6.X では，MS-DOS のほうがヘルプは詳細でしたが，PC-DOS Ver 7.0 ではさらに充実しました．従来の実行ファイルに **/?** を付けて得られる簡単なヘルプのほかに，次の方法により，詳細なヘルプ情報が得られるようになりました．

(1) Viewer による方法

拡張子が **.INF** である **CMDREF**(DOS のコマンド，**CONFIG** コマンド，デバイスドライバなど)，**DOSREXX**(REXX 情報)，**DOSERROR**(エラーメッセージ)があります．

(2) **Help** コマンドによる方法

Help にパラメータを付けて呼び出すと，パラメータが DOS コマンドの場合には **CMDREF.INF**，REXX コマンドの場合には **DOSREXX.INF** あるいはアルファベット1文字の場合は **DOSERROR.INF** の該当ページが開かれます(ESC キーで Contents の画面に戻る)．

また，不明確なときは，第1パラメータに **rexx** あるいは **error** を追加します．さらに，プログラムによっては，メニューバーのヘルプ項目を選ぶ，あるいは F1 キーを押すことによりヘルプが現れるものもあります．

● バックアップの充実

FILEUP.EXE(FileUpdate の意)というコマンドが導入されました．これは，二つ(あるいは同一)の PC の二つのディレクトリ間の整合性をとるためのコマンドです．LAN で接続された PC 間，INTERLINK で結合された PC 間の場合などにも対応しています．

離れた PC の場合には，ディスケットでファイルの交換をします．また，この操作から除外するファイルをその拡張子により指定できるようになっています．

Central Point Backup も従来どおりに供給されます．

また，**BACKUP.EXE** および **RESTORE.EXE** が再び，含まれるようになりました．

● DOSDATA

CONFIG.SYS コマンドに **DOSDATA** が追加されました．そして，**dosdata＝umb** と設定すると，**Files**, **Fcbs**, **Buffers**, **Lastdrive**, **Stacks** のシステムテーブルが，もし可能なら UMB にとられるようになりました．このため，640 K バイト未満のコンベンショナルメモリの残余量を大きくできるようになりました．

ただし，これは英語モードの場合の話で，DOS/V 用が将来出荷されたとしても，UMB 領域にそれほど余裕がないのが一般的ですので，このパラメータの恩恵

は少ないでしょう.

● システムエディタ E.EXE の改善

エディタもいくつかの点で改善されました. その主要なものをあげると,

(1) マウスが使用できるようになりました.

(2) 拡張キーボードのシステムで F11 および F12 が使用できるようになりました. 複数ファイルを扱うとき, F11 で直前のファイル, F12 で次のファイルに切り換えることができます.

(3) **BROWSE.COM** により, 読み取り専用で **E** エディタを使用できるようになりました.

(4) C および REXX のファイルを扱うとき, 構文の解釈によるキーワードの自動生成機能が付加されました.

● PC-DOS 6.3/V との併用

筆者は Ver 6.3/V からバージョンアップしました. Ver 6.3/V で立ち上げ, **SWITCH** コマンドで(注3)まず英語モードにします. マウスを使用できるようにしておいたほうがよいでしょう(インストールの過程でマウスが使えるようになる).

ここから, セットアップを開始します. ディレクトリは Ver 6.3/V のディレクトリと変えておいたほうが完全ですが, 同じにしました(結果的に, 同名のファイルは上書きされる).

日本語キーボードを使っているときは, **AUTOEXEC.BAT** の **KEYB.COM JP, 437, C:¥DOS¥KEYBOARD.SYS** などがインストール時に英語キーボード用に書き換えられますので, 元に戻す必要があります.

また, **KEYB.COM** および **KEYBOARD.SYS** は Ver 7.0 のもので上書きされますので, Ver 6.3/V のものに戻す必要がありました. この状態で, 本来の英語モードは 7.0 にバージョンアップできます.

SWITCH コマンドで日本語モードに戻してやると, 日本語モード用の **CONFIG.SYS**, **AUTOEXEC.BAT** は元のままですが, 日本語モードで立ち上がります. 使用していた DOS/V 用のコンフィギュレーションコマンド類は, 上記の **KEYB.COM** を除き, バージョンが違っても動くタイプでした.

かな漢字変換なども使用できるようになります. また, **CHEV.EXE** も使用できますので, 表示モードを一時的に日本語, 英語の間で切り換えることも可能です.

特に, PC-DOS 7.0 にはキャラクタベースのウィンドウ表示を用いたツールが多数付属しますが, これらは, 英語モードにする必要があります. 何れにしろ, **SWITCH** コマンドで英語モードと日本語モードの切り換えを行うこともできます.

インストールの手間を厭わないなら, PC-DOS Ver 7.0 用のディレクトリを別にすれば, 日本語モードをより完全なものにできるでしょう.

その場合には, 個々の同名のファイルについて, Ver 6.3/V と Ver 7.0 のいずれを使用するかの選択をし, Ver 6.3/V の不要なファイルを消去するようにすればよいでしょう. ただし, Ver 6.3/V 版を使うものの中には, **SETVER.EXE** によるバージョン登録が必要となるものが多量に発生するでしょう. また, 変則的な使い方ですので, 何等問題がないかは, 保証の限りではありません.

● その他の特徴

多数の新しいコマンド, あるいは改善されたコマンドがあります.

DYNALOAD.COM は, デバイスドライバを動的にロードすることができます. これは, 例えば NEC 用の DOS では **ADDDRV.EXE** として, かなり以前から実現されていたものです.

また, モービルコンピュータのホットあるいはウォームドッキングのためのプログラムとして, DOS 用の **DOSDOCK.COM** および Windows 用の **DDPOPUP.EXE** があります. **AUTOEXEC.BAT** で環境変数, **DOCK**, および **UNDOCK** で指定したコマンド(**.EXE**, **.COM**, **.BAT** ファイルなど)を接続およびその解消の時に実行することができます.

ディスク圧縮ソフトの Stacker V 4.0 への変更があります. これに伴い多くの新しいコマンドが導入されています.

また, コンベンショナルメモリの消費量を抑えるために, **COMMAND.COM** に新たに導入されたスイッチ **/H** は, **COMMAND.COM** の常駐部および環境変数領域をもし可能なら UMB にロードするためのもので, **EMM386.EXE** のパラメータ **MOVEXBDAHI** は, XBDA を, もし可能ならば, UMB にロードするという機能が追加されました.

(注1) 通常の 3.5 インチ 2HD が 18 セクタ×80 トラック×2 ヘッド＝総計 2880 セクタに対し, **XDF.COM** を介して覗いた XDF ディスケットは 23 セクタ×80 トラック×2 ヘッド＝総計 3680 セクタで扱われている. いずれも, セクタサイズは 512 バイトであるので, 通称, 前者を 1.44 M バイトと呼ぶのにならえば, 後者は 1.84 M バイトとなる. また, FAT のセクタ数は前者が 9 に対し後者は 11 である. なお, 実際のシステムディスケットでは, **XDF.COM** を常駐しないときのディスク構造のため, 5 セクタがバッドセクタとされている.

(注2) ただし, データ領域をコンベンショナルメモリの最上位にとる. 通常, **COMMAND.COM** の非常駐部は **9578:100**(**DOS=UMB** の時)あるいは **9579:100**(**DOS=NOUMB** の時)以降にロードされるが, **9D1E:0** あるいは **9D1F:0** 以降が上記データ領域となるので, **COMMAND.COM** 非常駐部が, 再ロードされるとき, その分, 下位に移動される. パラメータなしで, **XDF.COM** を常駐したときには, **XDF** の常駐後, **XDF** のプログラム部とデータ部に約 2 K バイトの空き(飛び地)ができるので, この無駄をなくすためのものと思われる.

(注3) **SWITCH.BAT** の機能は, 機種あるいはインストールの方法によって異なる. 必要なら適当に読み換えてください.

インターネットアクセス・レポート ⑤

《asustek.asus.com.tw》

● 最新の Pentium マザーボードの情報

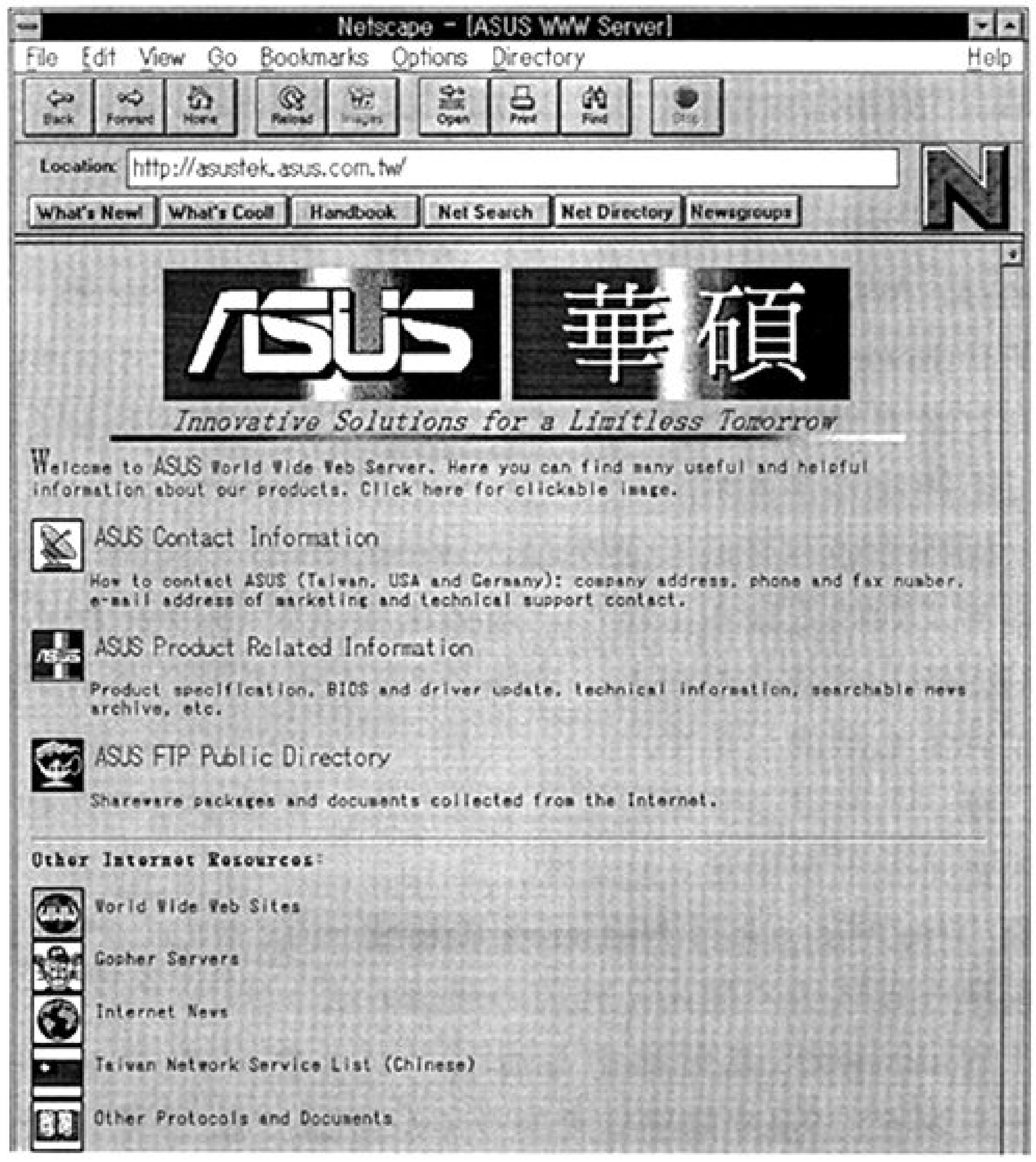
日本を含め，AT 互換機のマザーボードを設計製造しているパソコンメーカーは少なく，ほとんどのメーカーは，台湾で作られるマザーボードを利用しています．

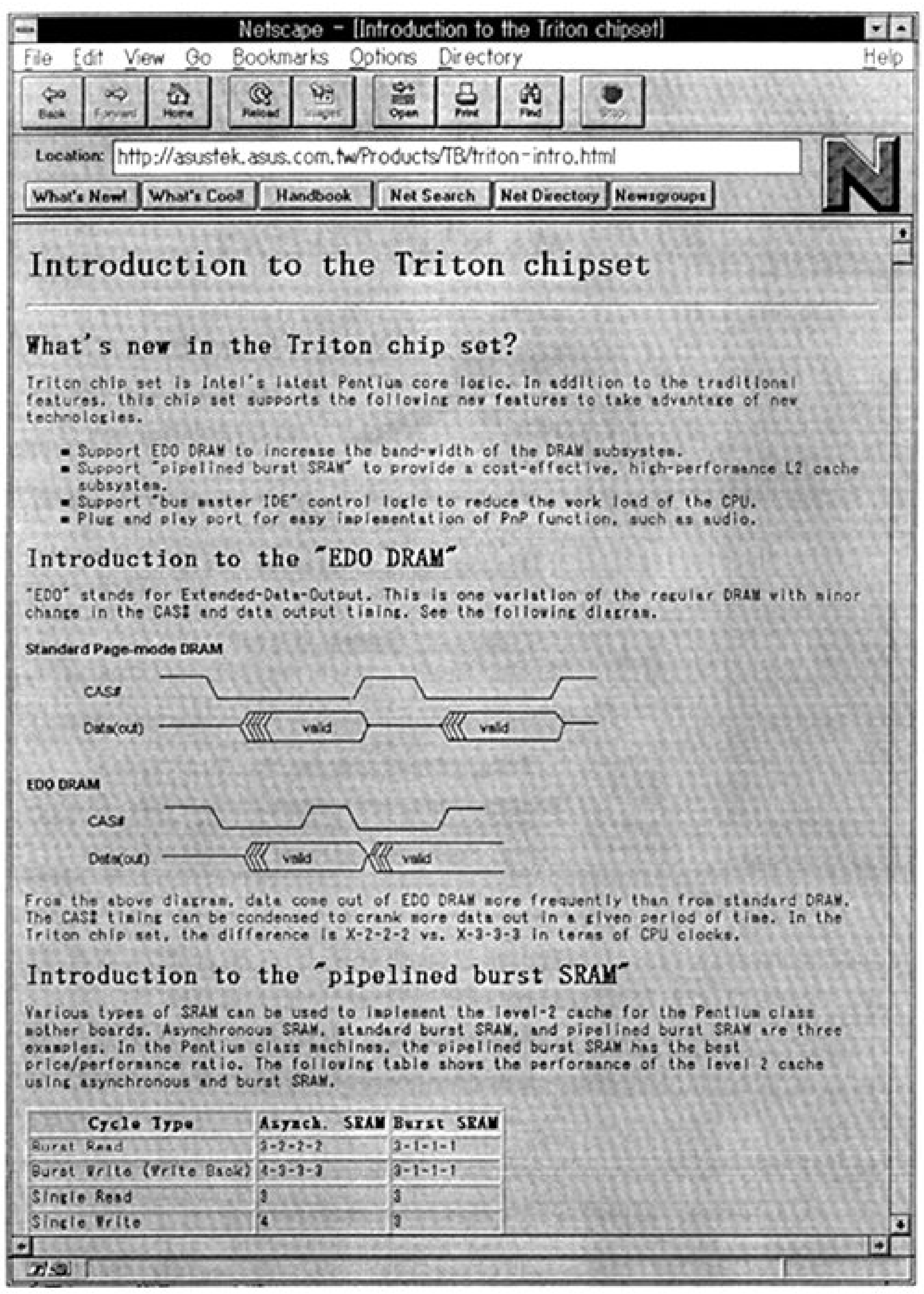
その中でもいくつか有名なブランドがあり，秋葉原でも購入できます．最近，Pentium CPU が 100～120 MHz になり，Triton チップセットの発表により，新規設計のマザーボードの出荷が始まっています．

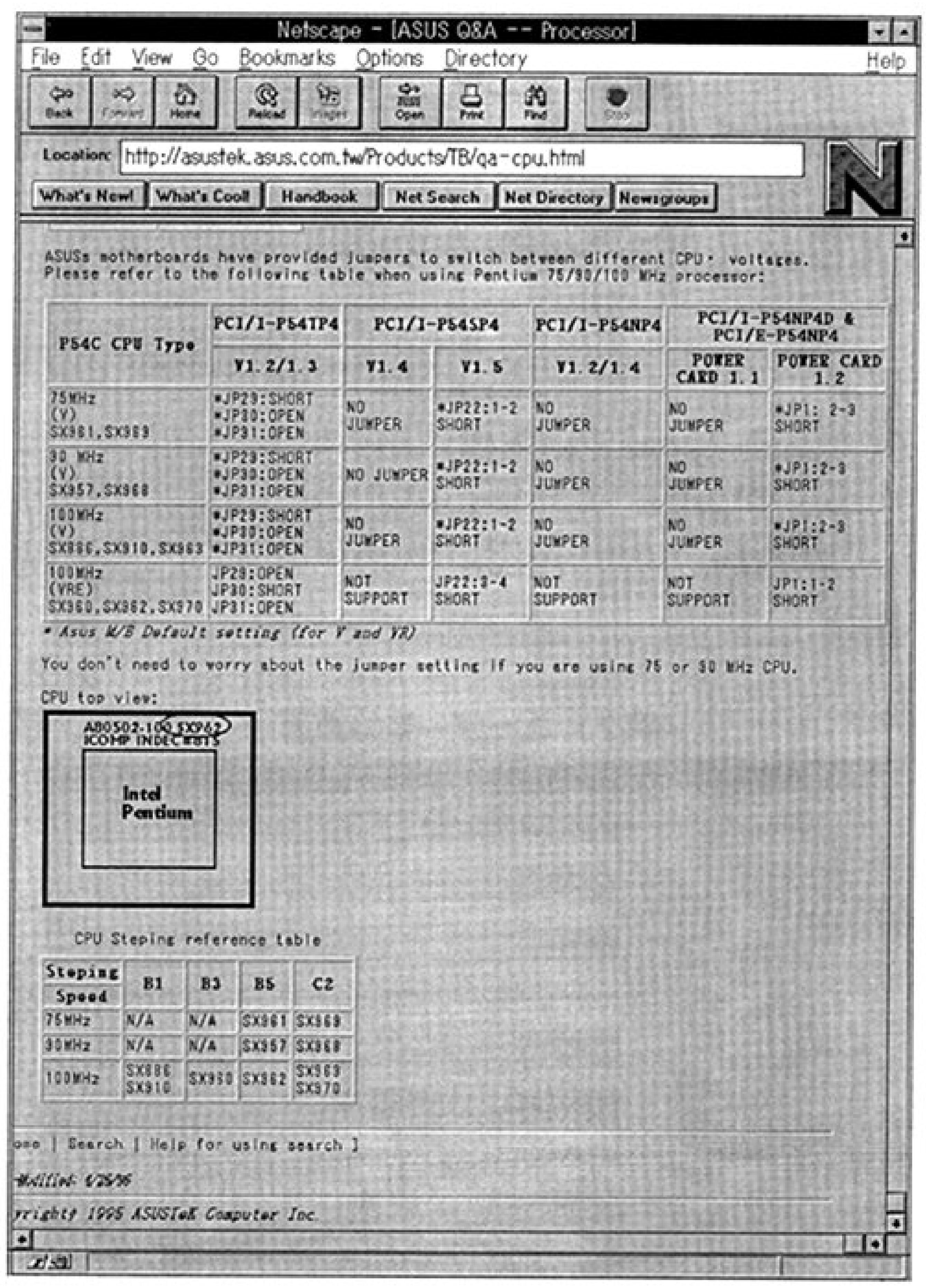
インターネット上でも BIOS のバージョンアップなどのサポートを行っている会社に ASUSTeK Computer 社があります．右の画面は，**asustek.asus.com.tw** に入ったところです．

マザーボードに関するサポート以外にも，Triton チップセットの特徴とか，Pentium CPU の 100 MHz には電源電圧が少し異なる (V；3.135～3.465 V，VR；3.3～3.465 V，VRE；3.45～3.6 V) モデルが存在しているなど，日本ではあまり入手できない情報が解説されています．

▼ ASUSTeK の WWW に入る







テクノロジエッセンス・レポート③

《BBSを始めるMicrosoft》

マイクロソフト社は，同社によるオンラインサービス「The Microsoft Network」の日本における試験運用を「Windows95」のベータ版配付に合わせて5/15から開始しました．

The Microsoft Network に接続するための通信機能は，Windows95の一部として含まれているため，Windows95を購入したユーザーはこのオンラインサービスを受けるためのアプリケーションを購入しなくても利用できます．

国内での本格的な運用は，Windows95の発売に合わせて開始される予定で，通信速度は2,400 bpsから14,400 bpsまで対応し，全国の11ヵ所にアクセスポイントを開設します．

サービスは全世界的規模で行われ，世界中にアクセスポイントが用意されます．

● The Microsoft Network の特徴

▶サービスへの加入と利用の開始方法

Windows95に付属する通信ソフトウェア(コンポーネント)により簡単にオンライン登録でき，すぐにサービスを利用できる．

▶コンポーネントマネージャ

The Microsoft Network のクライアントソフトのバージョン管理などは，サーバーから管理されるため，古いバージョンの場合は自動的に更新することが可能である．

▶二つの階層メニュー

サービス内容が一目でわかる「ホームベース」と呼ばれるビジュアルなメニュー．

▶電子メール

会員同士の電子メールサービス．Internet ともやり取り可能．

▶電子掲示板

メッセージのやりとり．

▶チャット

リアルタイム会議/マルチプレーヤゲームの提供．

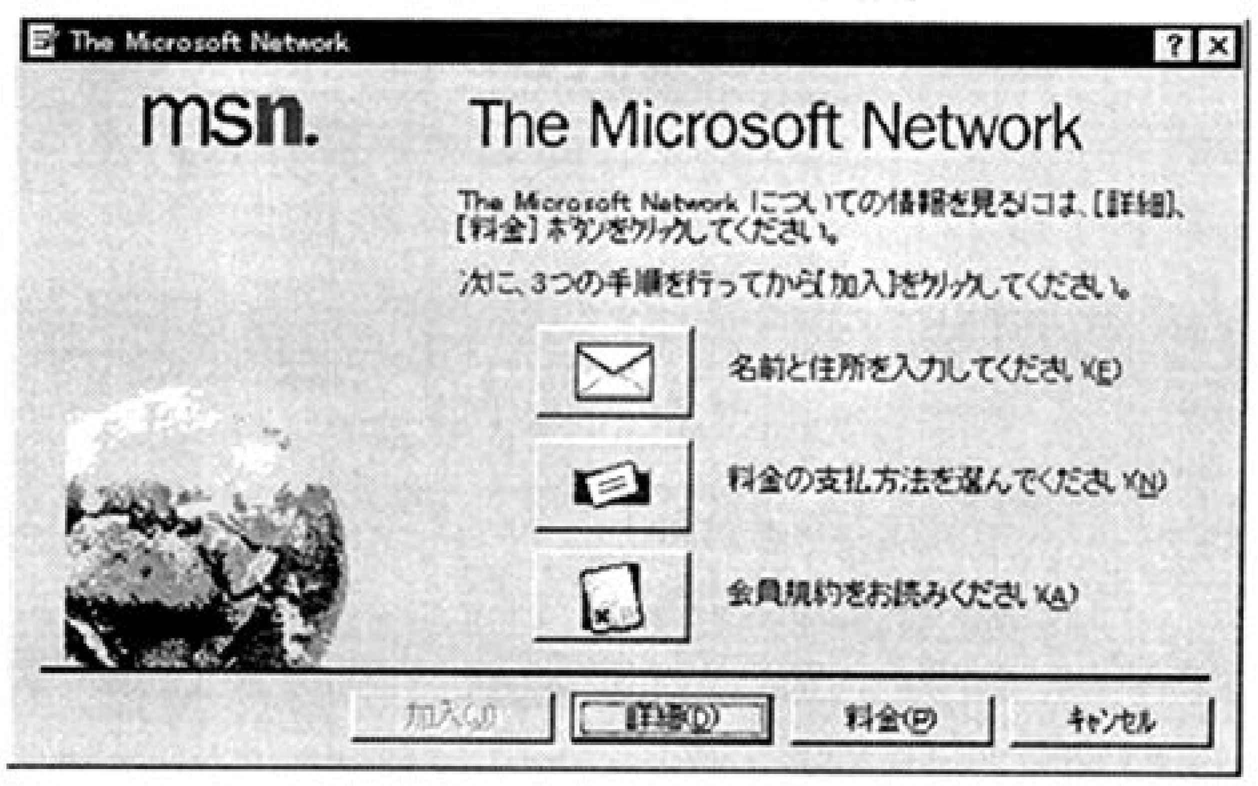
▶OLE2を利用したショートカットの利用

OLE2により，よく利用するサービスをデスクトップにアイコン化することができる．

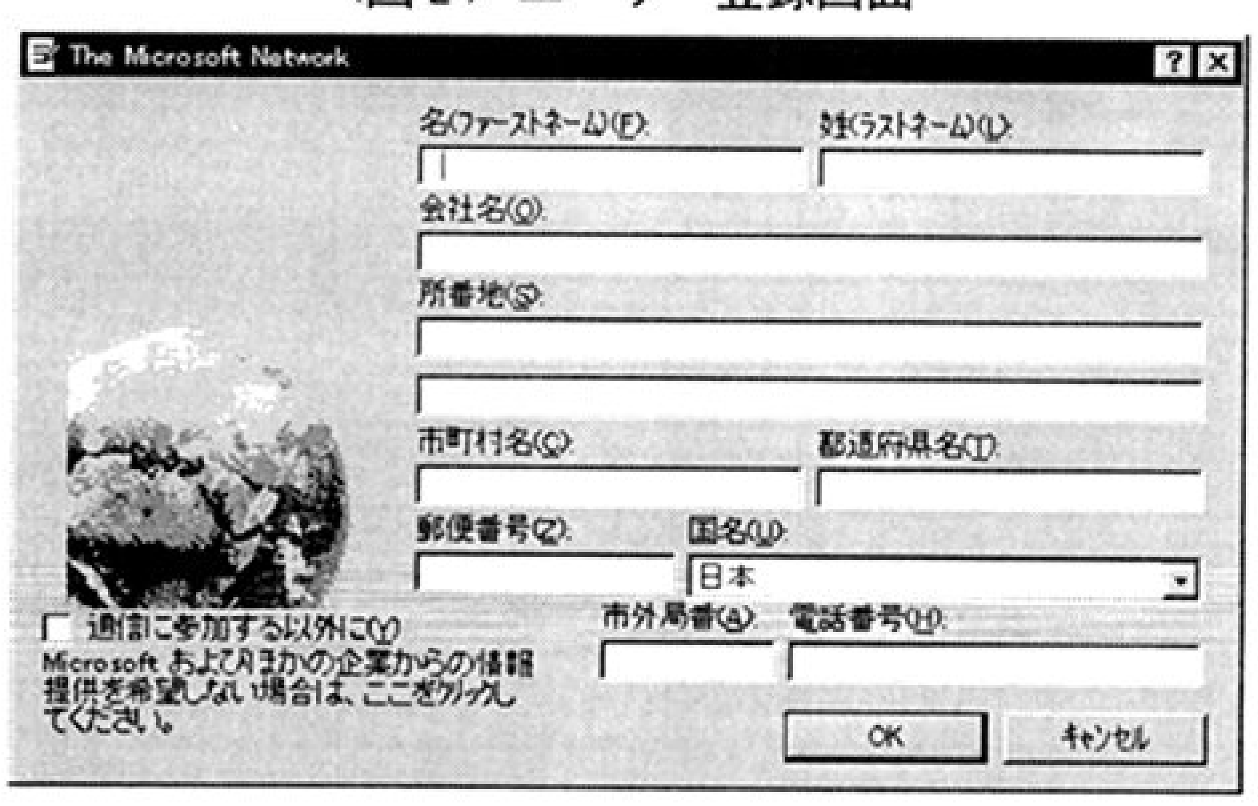
▶料金について

料金は未定だが，クレジットカードによる決済となる．

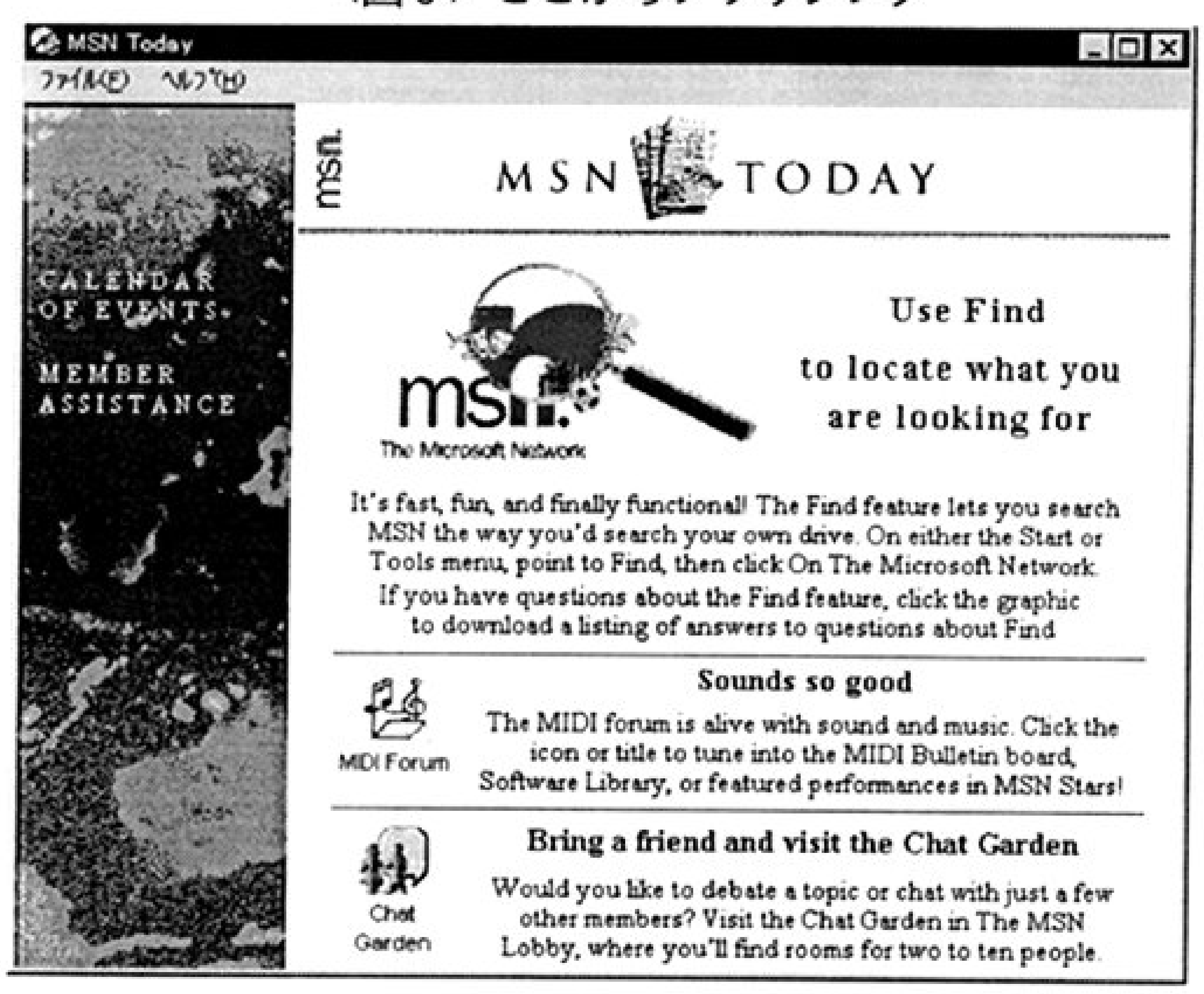
〈図1〉オープニング画面



〈図2〉ユーザー登録画面



〈図3〉ここからブラウジング



インターネットのWWWでは高速モデムを必要とするため，グラフィカルなユーザーインターフェースをもち，かつ14400 bpsでアクセス可能なBBSを作ったといえる．現在急成長中のAmerican Onlineの特徴とそっくりなコンセプト．サーバーには2CPUタイプのPCを140台接続してサービスをしているが，実用に耐えるレスポンスにいたっていない．8月のオープン時には高速化されているように期待される．

テクノロジエッセンス・レポート④

《Pentium に最適化された 3DR》

インテルジャパンは，3次元グラフィックスライブラリ 3DR に新機能を付加した 3DR Ver.2 を発表しました．3DR(Dimension Rendering)は，1994年4月に発表した同社のリアルタイム3次元グラフィックスライブラリで，Windows 上で高速描画を行うためのライブラリです．

3DR は，i486, Pentium に対応し，3次元グラフィックスやアニメーションなどの高速化を行うものですが，従来の機能に加えて以下のような機能が追加されました．

- ▶ 8,16 および 24 ビットモードを含むすべてのピクセル深度で 1677 万色を使用可能
- ▶画質とフレームレートの間でのトレードオフをバランスよく制御する X/Y ズーム機能
- ▶ P6 プロセッサへの移行が可能
- ▶ Windows 3.1, Windows 95, Windows NT との完全互換

また，3DR Ver.2 では，従来の 3DR/RE(ラスタライゼーション・エンジン)に加えて 3DR/GP(ジオメトリパイプライン)が加わりました．

3DR/GP(ジオメトリパイプライン)の機能は，次のとおりです．

- ▶一般光源モデルとマテリアルパレット
- ▶マトリクス操作とカメラモデル
- ▶テクスチャ処理ツールキット
- ▶ユーザーインターフェース・ツールキット
- ▶ Pentium プロセッサに最適化された数値演算ライブラリ
- ▶ Rational Quadratic Patches(RQP)のサポート
- ▶ Visual Basic のサポート

なお，従来の 3DR/RE(ラスタライゼーション・エンジン)の機能は次のとおりです．

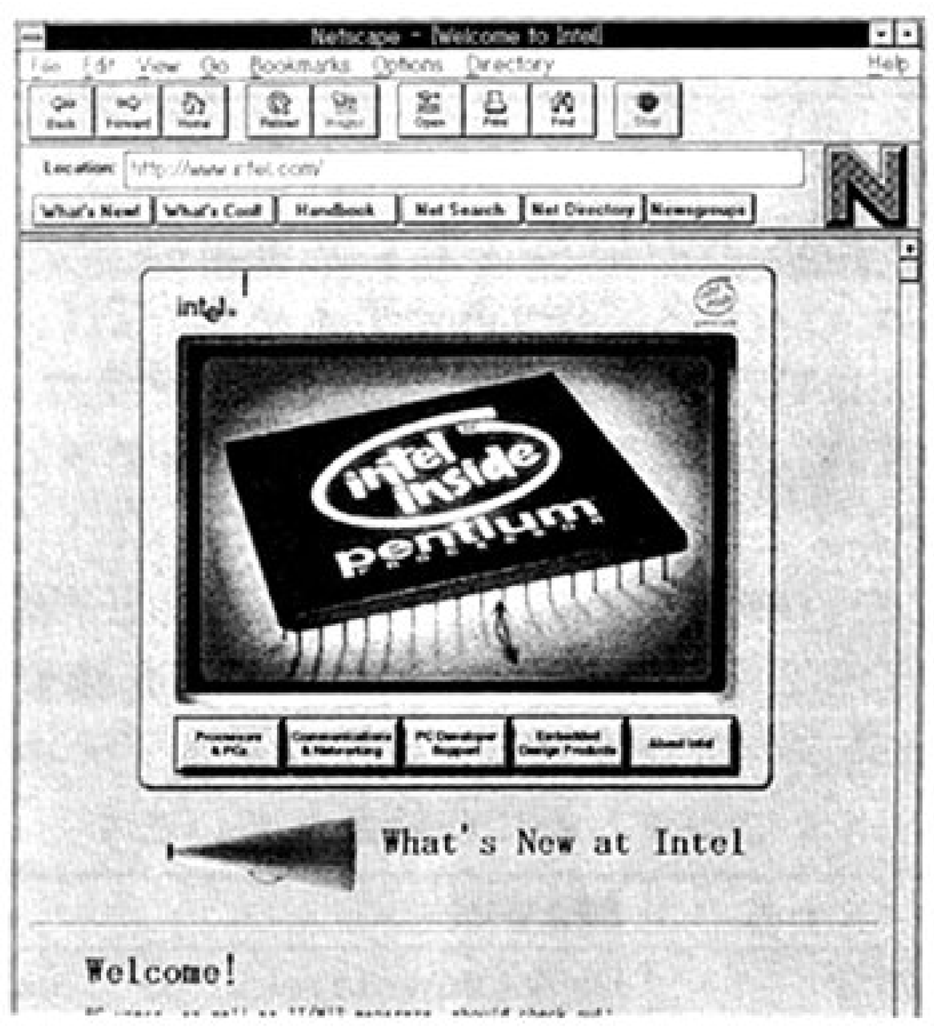
- ▶三角形，ポリゴン，ポリライン，線，点のレンダリング用3次元のプリミティブの装備
- ▶フォントとスプライトのラスターイメージおよびビットマップ

表示補正，フィルタリングおよびミップマッピングを含むマルチプルテクスチャマッピングアルゴリズム

- ▶ラスター操作，マスキング，パターン描画

3DR のランタイムは，ロイヤリティフリーで再頒布可能です．また，開発キット(SDK)については，下記へお問い合わせください．

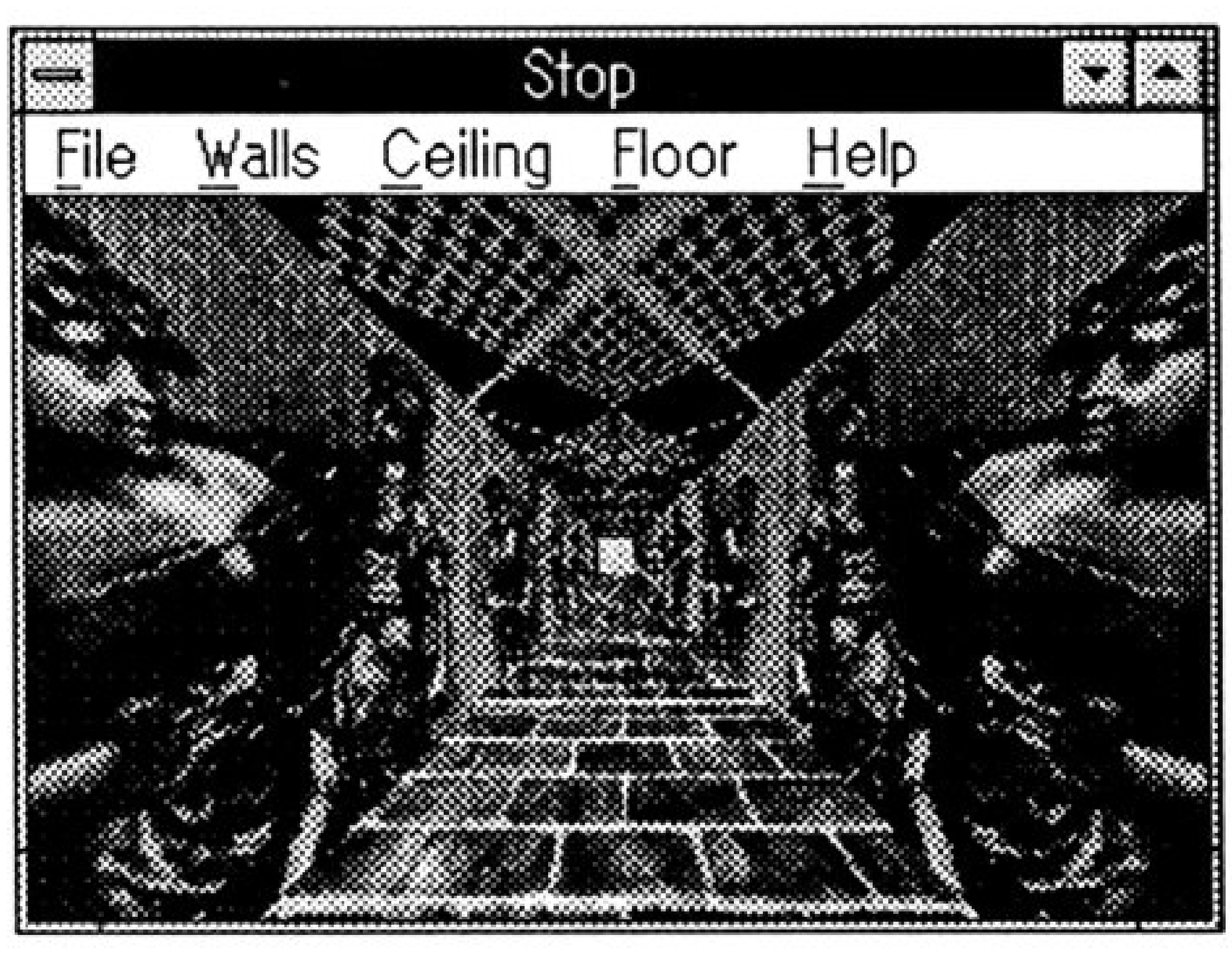
インテルジャパン
(株)プラットフォーム・マーケティング
☎ 0298-47-8511
Internet 上で情報を入手する場合
▶ **http://www.intel.com/**



〈図2〉
3DR を使った
サンプル ②
(迷路のサンプル)

〈図1〉 3DR を使ったサンプル ①
(マウスで内部をドラッグするとくるくる回転する)





CD-ROMレポート⑥

《Blowfish OS/2, OS/2 mate》

＊本稿で紹介したCD-ROMを1セットずつ読者に抽選でプレゼントします．希望される方は，アンケートはがきでご応募ください．

問い合わせ先▶パシフィック・ハイテック
〒154　東京都世田谷区梅丘1-14-4
☎03-5450-5003

OS/2 Warpの影響でしょうか，OS/2関連タイトルの動きが活発になってきました．

今回はOS/2シェアウェア集の中でも，最近発売されたBlowfish OS/2 '95.3(日本語ビューワ付き，**図1**)とOS/2 mate(**図2**)を題材に，便利なユーティリティを紹介します．

● **MLSW101**(Blowfish OS/2)

MLSW101はWindowsのAlt-Tabキーのような働きをする切り替えユーティリティです．Ctrl-Tabで画面の中央に起動しているアプリケーションや開いているフォルダなどが表示されます．

必要なところで離せば，そのアプリケーションに切り替わります．

ML Switcher-Settingを選べばMicroLearn Switcherの設定や，起動を止めることができます．

ソフト名	**MicroLearn Switcher**
作者	MicroLearn Nordic　jari@microlearn.se
分類	Freeware
動作環境	OS/2 2.X
動作確認	COMPAQ Prolinea MT 4/66，メモリ16Mバイト，OS/2 Warp
所在	**/DESKTOP/MLSW101/**

● **SEAWIN**(Blowfish OS/2)

SEAWINはOS/2上でWindowsのプログラムを高速にシームレスで行うユーティリティです．

SEAWINを起動するとバックグラウンドでWindowsを起動させるため，Windowsプログラムを高速に起動させることができます．しかも9Kバイトしかメモリを消費しないので，全体の動作にはほとんど影響がありません．

OS/2上でWindowsプログラムを頻繁に使用する方には最適のプログラムです．フォルダの中にはバージョン1～3までがまとめて収録されています．

ソフト名	**SEAWIN**
作者	Dr.A.Olowofoyeku(the Great Elephant) laa12@keele.ac.uk
分類	Freeware
動作環境	OS/2 2.X
動作確認	COMPAQ Prolinea MT 4/66，メモリ16Mバイト，OS/2 Warp
所在	**/DESKTOP/SEAWIN4/**

● **MEMO**(OS/2 mate)

MEMO ver.1.1は簡単なメモプログラムです(**図3**)．一つのデータファイルに最大255個のメモを書いておくことができて，さらに，各メモに対して指定した日時にアラームを鳴らすことができるので，予定を忘れてしまうような失敗もなくなります．

日本語でもメモを取ることができます．

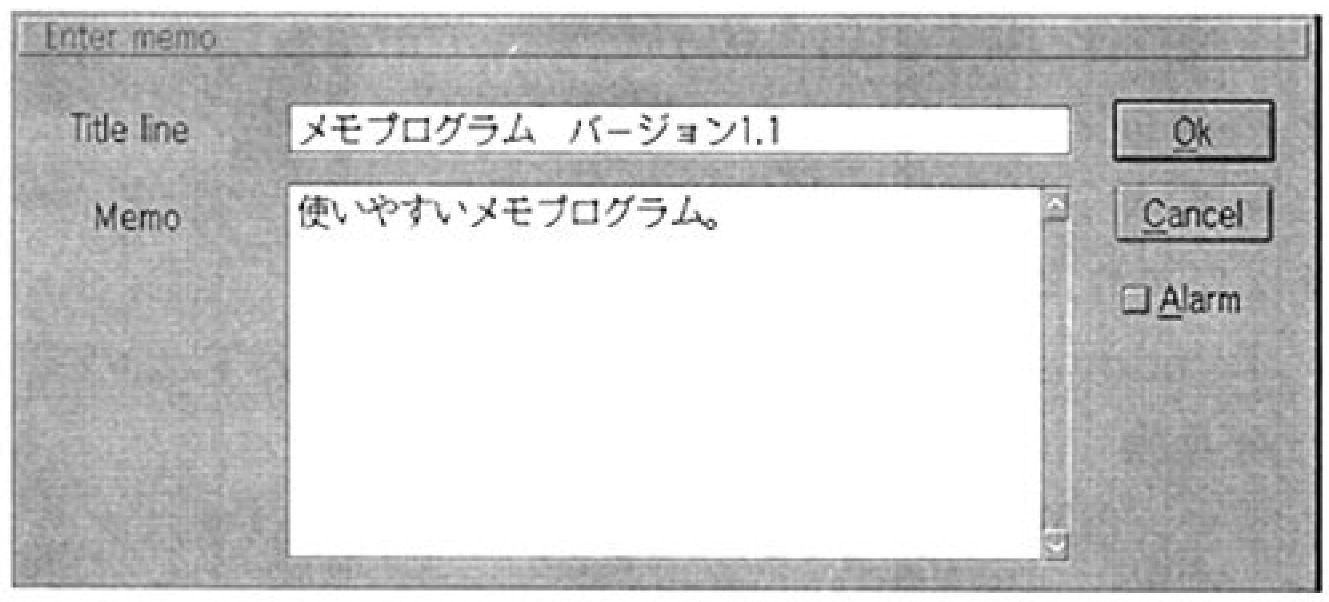
〈図1〉Blowfish OS/2 ▶





◀〈図2〉OS/2 mate

〈図3〉MEMO



ソフト名	**MEMO ver.1.1**
作者	Michael G. Hoodes/Sue Brown CIS：100115，357
分類/登録料	Shareware/$15
動作環境	OS/2 2.X
動作確認	COMPAQ Prolinea MT 4/66，メモリ16Mバイト，OS/2 Warp
所在	**/DESKTOP/MEMO11.ZIP**

● **その他**

▶ Blowfish OS/2

hippix：UNIXライクな開発環境，シェルコマンドを実現．**rexxdebu**：REXXのソースレベル・デバッガ．**run110**：DOSセッションからOS/2のプログラムを走らせる．

wdog10：デスクトップのハングを検知し，DOSシャットダウンを呼ぶ．**netbook8**：WWWサイト1325種を40分野に分類．

▶ OS/2 MATE

pmjpeg15.zip：JPEG，TIFF，GIF，TARGA，PCX，BMPビューワ．

OS2tnt18.zip：OS/2のヒントとテクニック集．**chk_er.zip**：IBMのCコンパイラ用エラーメッセージ・チェックプログラム．

alphal.zip：ソースコード・ブラウザ．**gccfat1~6.zip**：GNU CコンパイラのOS/2版．6枚組．

● マルチメディア・パソコン

マルチメディア・パソコンとは，ディスプレイやCD-ROMなどを最初から搭載した一体型パソコンを指す．

したがって，入門者には適しているが，拡張性に乏しいのが欠点である．CPUのスピードやハードディスクの容量などは初心者にとってすでに充分と思われるが，PC98やDOS/V機では今年の後半にはWindows 95が登場する予定になっている．入門者にOSの入れ替えを期待することは難しいため，現行のWindows 3.1インストールモデルの魅力が急速に薄れるかもしれない．その点，Macintosh PerformaはPowerPCモデルになり，入門者にも安心して薦められる．

すぐに使えるアプリケーションソフトウェアが最初からインストールされているのもマルチメディア・パソコンの特徴で，ほとんどに統合ソフトが導入されている．

松下電器のWOODY PDはCD-ROMドライブとしても使えるPD(相変化型光ディスク)ドライブを搭載し，TVや再生したビデオCDから取り込んだ画像データなどの大容量マルチメディア・データの記録・編集が可能になった．WOODYには辞書も搭載されている．

またこれらマルチメディア・パソコンのほとんどの機種にFAXモデムを搭載し，パソコン通信用ソフトも初めからインストールされているので，パソコン通信からFAXの送受信も気軽に行えるようになった．

コンパックのPRESARIOとエプソンのUG MULTIにはボイス機能も搭載されており，留守番電話やスピーカーフォンとしても使用できる．



▲ PRESARIO CDS524



▲ PC486ME



▲ Peforma5200



▲ CF-32GP

名称	情報家電 PRESARIO	EPSON UG MULTI	Macintosh Performa	WOODY PD
型名	CDS524/CDTV528	PC-486ME1LM/1LVM	Performa5210/5220	CF-32GP
CPU-MHz	486DX2-66	DX2-66 相当	PowerPC 603-75	DX4-100*
メモリ(最大)	8 M(64 M)B	8 M(128 M)B	16 M(64 M)B	8 M(32 M)B
HDD	420 MB	520 MB	500 MB	540 MB
グラフィック(ビデオRAM)	CL5424(512 KB)	拡張時 TGUI19660 (1 MB)	(1 MB)	9440AGi(1 MB)
ディスプレイ	14″CRT	15″マルチスキャンCRT	15″ マルチスキャンCRT	15″フラットCRT
標準搭載ハード	FAXモデム，4倍速CD-ROM，ステレオ・スピーカ，TVチューナ	4倍速CD-ROM，FAXボイスモデム，ステレオ・スピーカ，TV/MPEGボード	4倍速CD-ROM，FAXモデム，ビデオ入力カード，TVチューナ	PD/CD-ROMドライブ，MPEGデータ再生機能，FAXモデム，PCM/FM音源，DSP，TV機能
プリインストールソフト	MS-DOS 6.2/V，MS Windows 3.1，メディア・コントロールソフト「メディア・パイロット」，MS Works，PC PAINTBRUSHほか	EPSON MS-DOS 6.2，MS Windows 3.1，メニューソフト「パソピギランド」，MS Works，MS Golfほか多数	漢字Talk 7.5，InTernet Connect PPP・Mac Web，メニューソフト「At Ease 2.0」，クラリスワークス2.0ほか	DOS 6.2 V，Windows 3.1，TV/音楽CD/フォトCD/ビデオCDほかのビューワソフト，MS Works，国語/和英・英和辞書ほか
特徴	電話・FAX/電話・FAX・TVを統合したパソコン．	FAX送受信のほか留守番電話，電話	高速FAXモデムとインターネット接続および検索ソフトを標準装備	PDの内蔵により大容量画像データの記録・編集が可能
価格(円)	218,000/248,000	278,000/328,000	オープンプライス	318,000
問い合わせ先	コンパック・プレサリオ・サポートセンター ☎0120-099589	エプソンPCインフォメーションセンター ☎0424-99-7122，エプソンFAXインフォメーションセンター ☎0424-99-7100	アップルカスタマーアシスタンスセンター ☎0120-61-5800，FAXAID：03-3391-1200，06-212-7111，プレスリリース：1802#，カタログ：2801#	松下電器産業㈱ WOODYサポートセンター ☎03-3438-2820，FAX情報サービス ☎03-3940-6000 (BOX番号：875604)

*：オーバードライブ対応

〈略号〉KB：キロバイト，MB：メガバイト，GB：ギガバイト，
HDD：ハードディスク・ドライブ，FDD：フロッピディスク・ドライブ

● パーソナルコンピュータ

マルチメディア・パソコンやノート型パソコンが急成長しているので，従来からのパソコンは，高性能と拡張性を競うことになり，タワー型も珍しくなくなった．

NECは98MATEを一新し，98MATE Xとして発売した．全機種Pentium，PCIバス，4倍速CD-ROMを搭載したのに加えて64ビットWindowアクセラレータチップ，2台目のハードディスク専用ベイの装備により最大1.2Gバイトまで追加できるディスク容量とマルチメディアへの対応力も強化した．

基本ハードウェアの大きな変更点としては，D-sub25ピンのシリアルインターフェースのチャネル1が転送レート115,200bpsに拡張され，さらにIBM PCで採用されているUART16550互換のシリアルインターフェース(チャネル2，D-sub9ピン，転送レート115,200bps)が追加された

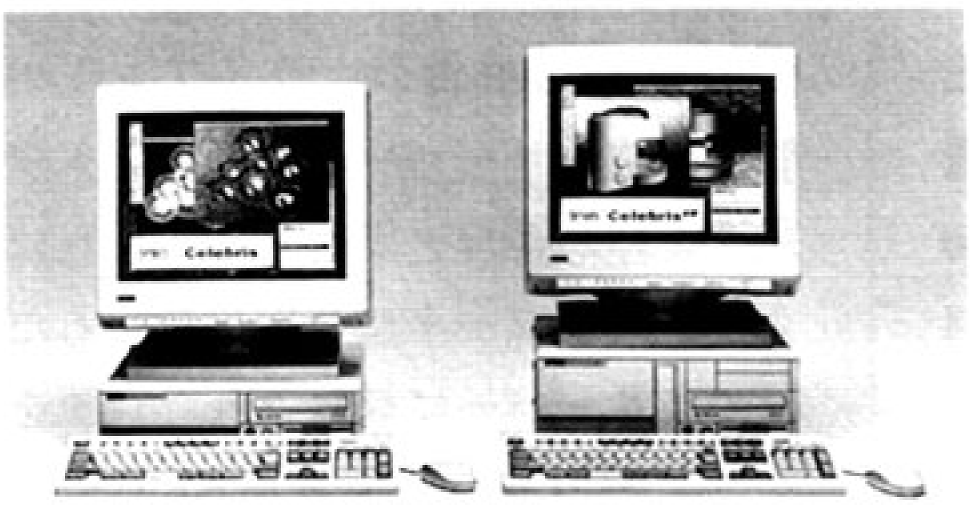
コンパックのDESKPROシリーズは，PCIバス対応の32ビット・イーサネット・インターフェース(10BASE-T/5)を標準内蔵し，ネットワークドライバ，NetWare/LAN Managerクライアントソフトを標準インストールしたネットワークパソコン．

日本DECのCelebrisXLは，システムアップの柔軟性を考慮したCPUドータカード方式を採用．また，ソードもDOS/Vパソコンの発売に乗り出した．BT5000以外のハードディスク・モデルにはすべてDOS 6.2/V(NECはDOS 6.2)，Windows 3.1がインストール済み．

名称	DESKPRO 590/5, 5100/5	ProLinea 5100/5	CELEBRIS		
型名	M720QV	M720	Celebris 575	CelebrisFP 5100	CelebrisXL 5100/5100P
CPU-MHz	Pentium-90, -100	Pentium-100	Pentium-75	Pentium-100	Pentium-100/Pentium-100×2
メモリ(最大)	16M(192M)B	←	8M(128M)B	16M(128M)B	16M(192M)B
拡張バススロット	PCI×1, PCI/ISA×1, ISA×3	←	PCI×1, PCI/ISA×1, ISA×1	PCI×1, PCI/ISA×1, ISA×3	PCI×2, PCI/ISA×1, ISA×3
HDD	720MB(IDE)	←	540MB	←	1GB(SCSI)/オプション
グラフィック(ビデオRAM)	Qvision 1280P+(1MB)	GD5434(1MB)	Vision864(2MB)	←	Vision964(Stealth64, 2MB)
特徴	PCIイーサネット(10BASE-T/5)内蔵，NetWare/LANマネージャ・クライアントソフト標準インストール	TriFlex/PCIアーキテクチャ採用	2倍速CD-ROM搭載タイプ	2倍速CD-ROM搭載，拡張性に優れる	3.3倍速CD-ROM(SCSI)，オンボードにSCSI-2コントローラ搭載，CPUドータカード方式採用，5100PはWindowsNT 3.5のSMP対応
備考	ミニタワー型，ネットワークPC，Windows95対応	ミニタワー型，Windows95対応	薄型デスクトップ		ミニタワー型，Plug&Play対応
価格(円)	438,000, 478,000	448,000	278,000/298,000	438,000/458,000	698,000/798,000
問い合わせ先	コンパック・カスタマーセンター ☎0120-101589		日本ディジタル イクイップメント(株) Digital PCホットライン ☎0120-147655		

名称	INNOVA5590	デスクトップDOS/Vパソコン	98MATE X		
型名	INNOVA5590-S/540/850PW	BT5000/FD, /WIN	PC-9821Xa10/C12	PC-9821Xa9/C8	PC-9821Xa7/C4
CPU-MHz	Pentium-90	486SLC2-50	Pentium-100	Pentium-90	Pentium-75
メモリ(最大)	0/8MB/8M(128M)B	1MB/8M(16M)B	16M(128M)B	8M(128M)B	8M(128M)B
拡張バススロット	PCI×3, PCI/ISA×1, ISA×3	ISA×2	汎用×3, PCI×2, PCカード×1	汎用×3, PCI×2	←
HDD	オプション/540MB/850MB	なし/420MB	1.2GB	850MB	420MB
グラフィック(ビデオRAM)	オプション/AT1264(1MB)/PowerWindow864(2MB)		TGUI9680XGi(2MB)	←	←
特徴	Tritonチップセット搭載のミニタワー型	モノクロ/カラー9″CRT付属のノートサイズ	4倍速CD-ROM搭載	←	←
備考		付属のソフトはない，キーボードは101相当	デスクトップ型，Windows95に対応	←	←
価格	198,000/298,000/368,000	68,800/148,000	498,000	398,000	268,000
問い合わせ先	キヤノン販売(株) ☎043-211-9431	BookTopダイレクト，〔(株)アドテックシステムサイエンス〕 ☎045-337-3723, FAX 045-331-7770	パソコン情報サービス FAX 03-3552-9801 BOX番号：7049 NECパソコンインフォメーションセンター ☎022-262-9800		

〈略号〉KB：キロバイト，MB：メガバイト，GB：ギガバイト，HDD：ハードディスク・ドライブ，FDD：フロッピディスク・ドライブ



▲ Celebris



▲ 98MATE X



▲ ThinkPad 530CS

● カラープリンタ

名称	フルカラー・ページプリンタ	カラーインクジェットプリンタ
型名	BJC-680	PC-PR101/J180
解像度	360 dpi	モノクロ：600×600 dpi，カラー600×300 dpi（専用紙の時）
カラー印刷	四色一体型ヘッド	←
内蔵フォント	平成明朝体W3，平成角ゴシック体W3，欧文書体(Swiss, Dutch, Symbol)	明朝体，ゴシック体，毛筆体
用紙サイズ	普通紙A4，B5，レター，ハガキ	普通紙A4，B5，A5，レター，ハガキ，封筒
給紙方式	オートシートフィーダ，手差し	オートシートフィーダ
インターフェース	セントロニクス準拠，RS-232C準拠（自動切換可能）	セントロニクス準拠
コマンド制御	LIPS IVc/III，ESC/P24-J84(VP-3000)，キヤノンBJラスタイメージコマンド	201PL，ESC/Pエミュレーション
価格	248,000	64,800
問い合わせ先	キヤノン販売㈱BJコールセンター ☎03-3455-9544	NECパソコン情報FAXサービス ☎03-3552-9801(BOX番号：7040)，NECパソコンインフォメーションセンター ☎06-943-9800

● ノート型パソコン

ノート型パソコンもDX4やPentium CPUなどを採用した高性能機種が当たり前になってきた．また，ローカルバスにグラフィックチップを接続して，512 Kあるいは1 MバイトのビデオRAMを搭載した機種が増えてきた．

ThinkPad 530CSはマルチメディア・パソコンと同じように高速のFAXモデムを備え，パソコン通信，FAX，電話などができる．

さらにThinkPadは，同一ハードディスク上にOS/2 Warp，PC DOS J 6.3 V，Windows 3.1を初期導入しておき，アイコンをクリックするだけでOS/2 WarpとDOS/Windowsの環境を自由に選択できるセレクタシステムを採用した．このシステムは今後540 Mバイト以上のハードディスクを備えるIBM PC，ThinkPad，Aptivaの新製品に標準採用される．

● カラープリンタ

キヤノンのBJC-680は，ページ記述言語LIPSのカラーバージョンを搭載したモデル．Windows環境においてフルカラー印刷ができる．A4標準原稿のランニングコストが18円と安価なことが特徴．

NECのPC-PR101/J180は，ヒューレットパッカードのOEM製品で従来の同価格帯の製品に比べ，高画質，高速印刷が可能になった．

名称	ThinkPad		FMV-BIBLO		
型名	530CS	370C	FMV-575NA/X	FMV-575NA/T	FMV-499NA/T
CPU-MHz	SLエンハンスド486DX2-50	DX4-75	Pentium-75	←	DX4-100
メモリ(最大)	4 M(20 M)B/12 M(20 M)B	4 MB/←/12 M(36 M)B	8 M(72 M)B	←	←
HDD	540 MB	340 MB/540 MB/←	810 MB	510 MB/810 MB/810 MB	510 MB/810 MB
ディスプレイ	8.4"DSTN，640×480 256色	10.4"TFT，640×480 256色	10.4"TFT，1024×768 256色	10.4"TFT，640×480 65536色	←
グラフィック(ビデオRAM)	(512 KB)ローカルバス	(1 MB)ローカルバス	F65545(1 MB)	←	←
サイズ	241×180×41	297×210×51	299×261×58	←	←
重量	1.7 kg	2.9 kg	4.1/4.2 kg	3.9/4.0/4.1 kg	3.9 kg
バッテリ	NiMH	←	Duracellパワースマートバッテリ DR15(残量表示機能付き)	←	←
標準装備品	Mwave DSPチップ，14.4 Kbpsデータ/FAXモデム，留守番電話機能，DSP，マイク/スピーカ，PCカードスロット	PCM音源，スピーカ，PCカードスロット	サウンド機能，PCカードスロット，2倍速CD-ROM搭載(FDDと内蔵時排他)機種も	←	サウンド機能，PCカードスロット
価格(円)	オープン価格	478,000/528,000/628,000	958,000/998,000	668,000/718,000/758,000	598,000/648,000
備考	PC DOS J 6.3 V，Windows 3.1，OS2 Warp V 3，セレクタシステム	PC DOS J 6.3 V，Windows 3.1，OS2 Warp V 3，セレクタシステム	Windows95対応可能，DOS 6.2/V，Windows 3.1	←	←
問い合わせ先	ダイヤルIBM ☎0120-04-1992		富士通㈱FMインフォメーションサービス ☎03-3646-0816		

コダック，DC40
コダック デジタルカメラ 40

DC40は，コンパクト・カメラ感覚で使える普及型のデジタル・カメラ(114,800円)．写真フィルムの代わりに約38万画素のCCDを用いてディジタルで記録する．コンピュータで直ちに処理できるので速報性に優れ，画像の保存・加工・高速伝送などが容易．RGB各8ビットのフルカラー画像，単3リチウム電池×4で連続約800コマの撮影が可能．TWAINモジュール付属．Mac，Windows対応．

▶日本コダック㈱ デジタルイメージング事業部 ☎03-5488-2920



オムロン，日本語ペン型OCR Peeo V(ピーオ ブイ)
Windows対応 活字文章ピックアップツール

ペンスキャナによるハンドスキャン方式で，活字文章をハサミで切り取る感覚で必要な部分だけパソコンに入力できるOCR(39,800円)．RS-232Cインターフェースを採用し，PC98，DOS/Vのいずれにも使用可能．「知子の情報/Windows」機能縮小版添付によりメモリに蓄積した文章を検索できる．

▶オムロン㈱ 情報機器事業部 ☎045-411-7225



ランドコンピュータ，フロッピーコピア(LMC-35UG)
フォーマット変換機能付きメディアコピー機

異機種コンピュータ間の，サイズもフォーマットも異なったフロッピディスクをコピーできる装置(280,000円)．3.5インチ(2DD：640KB・720KB，2HD：1.2MB・1.44MB)と5インチ(2DD：640KB・720KB，2HD：1.2MB)のドライブを各2基搭載し，コンピュータは不要で，対話メニューによる簡単な操作でコピーできる．デュプリケーション機能，ファイルコンバート機能を装備．

▶㈱ランドコンピュータ ☎06-304-8403，FAX情報BOX ☎06-304-6959 LMC-35UGカタログ：5212



日本電気，AtermIT30
高速ISDNターミナルアダプタ

非同期57.6kbps通信を実現し，インターネットに不可欠なMosaicなどを利用したWWWサーバアクセスや，NetWare Connect，WindowsNT・RASなどによるLAN間接続，遠隔地のPC LANへのリモートアクセスなどがより快適に利用できるほか，コストも低減できる．自動速度整合機能，速度指定発信機能，フロー制御機能などにより既存のTAやネットワークとの接続性にも配慮．INS 64のほか，高速ディジタル回線Iインターフェース(64kbps)にも対応可能．

▶NECパーソナルコミュニケーションインフォメーションセンター ☎0120-361138

日本電気・富士通，NetWare 4.1J
ネットワークオペレーティング・システム新バージョン

NDS(NetWare Directory Service)の採用により，1台のサーバーでの最大1000台のクライアントの管理，複数サーバーを含むネットワーク全体の管理を実現した．これによりネットワークに一度接続するだけで各サーバーに登録された資源をすべて利用できる．またSFT IIIを導入すればサーバーの二重化が可能で，運用性・信頼性の高いシステムを構築できる．5ユーザー～1000ユーザー：190,000～6,240,000円．

▶NECパソコンインフォメーションセンター ☎03-3452-8000，

▶富士通㈱ システムサポートセンター ☎0120-009940

ボーランド，dBASE 5.1J for Windows
dBASE Windows対応版の最新バージョン

dBASE 5.0J for Windowsにクライアント/サーバー機能を追加したもの(98,000円)．ボーランド製品共通の「Borland Database Engine」の最新版を搭載しており，Paradox for Windowsなど，ほかのボーランド製品とのデータ互換性を実現したほか，Borland InterBaseやOracleなどの各種SQLデータベース・サーバーとの接続機能をもつ．また，dBASE for Windowsをパワーアップするツール集「dBASE for Windows Power Pack」(14,800円)も発売した．

▶ボーランド㈱ インフォメーションセンター ☎03-5350-9380

IBM，OS/2 Warp Connect V3日本語版
OS/2 Warpに接続ソフトを付加

OS/2 Warpファミリの一つで，LANや遠隔地接続などの多彩な接続ソフトを備えたもの．必要な機能を選択するだけで導入できるため，ネットワーク環境を容易にまた経済的に構築できる(26,000円)．含まれているソフトは以下の通り．IBM Peer for OS/2 V1.0，OS/2 LANサーバー リクエスタ，LAN Distance リモートJ1.1，TCP/IP for OS/2 V3，NetWare Client for OS/

2 (J)IBM版，Network SignON Coodinator (NSC)など，ほかに，これらのソフトを利用する際の問題解決用データベースおよび検索用ソフトと，Worksなどの身近なアプリケーションの入ったボーナスパック，OS/2 Warp Connect V3 with WIN-OS/2日本語版(33,000円)も発売．8月ごろから出荷予定．

▶ダイヤルIBM ☎0120-04-1992

日本コダック，Windows版 Kodak Arrange-It
フォトCDポートフォリオ レイアウト ソフトウェア

従来，Macintosh上でのみ行われていたポートフォリオ作成のレイアウト作業をWindows上でもできるようにしたソフト(84,900円)．フォトCD画像やBMP，TIFFのファイル形式で保存された画像，WAV形式のオーディオファイルなどを取り込み，画面上のアイコンを選択してオーディオクリップを画像にリンクさせることができる．パスの指定，タイムアウト，トランジション，イメージパックの設定などレイアウト機能をすべて制御できる．ベリファイコマンド，エミュレーションコマンドなどを利用して画面上で再生の確認が可能．

▶日本コダック(株) デジタルイメージング事業部 ☎03-5488-2920



松下情報システム，ウェーブデザイナー
インターネットサーバー用簡単入力ソフト

インターネット用WWWサーバーへ従来の5倍の速さで情報を入力でき，HTMLを知らなくてもマウス操作で入力画面を作成し，自動でHTMLデータを生成できるソフト．「イメージデータ登録」ソフトにより登録されたイメージを，「ページデータ編集」ソフトにより編集し，「サーバーデータ生成」ソフトによりHTMLを生成し，「サーバーデータ登録ツール」ソフトによりWWWサーバーに転送する．オプションで検索機能と検索結果の表示機能が付加できる．UNIX対応版：100万円，WindowsNT対応版：50万円．

▶松下情報システム(株)

☎06-350-3744，FAX 06-350-3750

日本コダック，Macintosh用 Kodak Build-It
フォトCDポートフォリオ ディスクへの書き込み用ソフト (日本語版)

従来，UNIX搭載のサンワークステーション用だけだった書き込み用ソフトのMacintosh版で，機能はUNIX版とほぼ同じ．ISO 9660のレベル1/2をサポート，マウスを使用したドラッグ&ドロップによる簡単な操作に加え，「Kodak Arrange-It」と組み合わせれば編集から書き込みまで1台のコンピュータで実現できる．フォトCDイメージパック作成だけの使用も可．「PCDライター225」および高速SCSIボード，専用SCSIケーブルを組み合わせたコンプリートキット(700,000～750,000円予定)で販売される．ソフトウェア単体は140,000円．

▶日本コダック(株) デジタルイメージング事業部 ☎03-5488-2920



ジャストシステム，一太郎 Ver.5 for Macintosh
一太郎の初めてのMacintosh対応版

Macintosh，PowerMacintosh対応版で，「一太郎 Ver.5 for Windows」とほぼ同等の機能を実現したほか，プルダウン・メニューのコマンド体系やキー割り付けなどをMacintoshのユーザーインターフェースに準拠した．58,000円．限定10万本の新発売キャンペーン(30,000円)や，3ヶ月間他機種版ユーザー優待(18,000円)，他社ワープロからの乗り換え(25,000円)キャンペーンなど．

▶(株)ジャストシステム インフォメーションセンター ☎0886-52-8555，☎03-5412-3939，☎06-390-6440

インターコム，まいと～く Ver.3
MS-DOS版統合型通信ソフト

PC98ユーザー向けに，パソコン通信機能とパソコンFAX機能の両方を搭載した通信ソフト(28,000円)．パソコン通信機能に加えて，パソコンから電話回線を介して直接FAX送信できる機能を新たにサポートした．同報通信，ファックスオンデマンドにも対応．また全面的にGUIを採用したことにより操作性が飛躍的に向上した．「セレクター」のサポートにより，アイコンをマウスでクリックするだけのビジュアルオペレーションを実現，まいと～くの標準機能のほか市販ソフトを登録して直接起動することもできる．

▶(株)インターソフト 営業部

☎03-3842-6022



ウイルス・レポート ㊾
〈Jimi〉

ワクチン入手先 ▶スキャンワクチン：9,800円
〒105 東京都港区虎ノ門3-7-2 ☎03-3435-1348
〒420 静岡県静岡市常磐町3-6-11
株式会社 ジェード ☎054-252-0085

このウイルスは，ハードディスクのマスタブート・レコードとフロッピディスクのブートセクタに感染する **Jimi** というブートセクタ・ウイルスで **NASITA** とも呼ばれています．IBM機特有の割り込み処理を使用しているため，IBMおよびその互換機以外で感染することはありません．

このウイルスに感染しているフロッピディスクからマシンを起動すると，まず，ハードディスクのマスタブート・レコードに感染します．ウイルスは，ハードディスク感染後メモリに常駐し，フロッピディスクへのアクセスを監視します．

ウイルスがメモリに常駐している状態でフロッピディスクにアクセスすると，アクセスしたフロッピディスクにウイルスが感染し，広がります．ウイルスは，すでに自分自身が感染しているかどうか調査していますので，重複して感染することはありません．

このウイルスは，11月15日にこのウイルスに感染しているディスクからマシンを起動した場合に発病し，ハードディスクをフォーマットして破壊することを目的としています．ただし，現在多く使用されているハードディスクでは，このウイルスによってハードディスクがフォーマットされ，破壊されることはありません．

このウイルスは，自分自身を暗号化することにより，ウイルスの露見を防止しています．

ディスクのルートディレクトリに多くのファイルが存在しているフロッピディスクにウイルスが感染した場合，ファイルの情報が破壊され，**DIR** コマンドでディスクを確認すると，意味不明なファイル名が表示されます．

ウイルス名称	Jimi		
ウイルス俗称	NASITA		
起源	不明		
国内発見日時	1995年		
ステータス	NEC	IBM	富士通
▶感染力	□□□□□	■■■□□	□□□□□
▶露見しにくさ	□□□□□	■■■□□	□□□□□
▶被害	□□□□□	(小)■■□□□(大)	□□□□□
タイプ	▶LMB常駐型 ▶起動時汚染型 ▶ディスクアクセス感染型 ▶BOOT寄生型 ▶暗号型		
ウイルス長	▶メモリ常駐 ……………2,048バイト ▶ディスク内 ……………512バイト＋512バイト		
OS	MS-DOS，IBM-DOS		
症状	ディスクのルートディレクトリに多くのファイルが存在しているフロッピディスクにウイルスが感染した場合，ファイルの情報が破壊され，**DIR** コマンドでディスクを確認すると，意味不明なファイル名が表示される．		

● **確認方法**

以下の症状が確認された場合は，このウイルスに感染している可能性がありますが，このウイルスであることを確認するためには，ワクチンソフトを使用してください．

▶ディスクのルートディレクトリに多くのファイルが存在しているフロッピディスクにウイルスが感染した場合，ファイルの情報が破壊され，**DIR** コマンドでディスクを確認すると，意味不明なファイル名が表示される．

▶メモリ容量が通常よりも2Kバイト減少している．

● **復旧方法**

まず，マシンの電源を切り，感染の恐れのないDOSのシステムディスクでマシンを起動します．

▶ハードディスクの修復

FDISK コマンドの/**MBR** オプションを使用してハードディスクを再構築することで，ウイルスを削除し修復します．

ただし，**FDISK** の/**MBR** オプションは，DOSのバージョンによって使用できないため，このコマンド実行後，ウイルスが削除されたことを確認する必要があります．

▶フロッピディスクの修復

フロッピディスクを **FORMAT** コマンドで初期化してウイルスを削除し，修復します．DOSのVersion 5以上を使用している場合は，**FORMAT** コマンド使用時に/**U** オプションを使用し，完全に初期化してください．

フロッピディスクのデータは，あらかじめほかの感染していないディスクに **COPY** コマンドでバックアップしておくことが必要です．

● **追加パラメータ**

以下はこの **Jimi** ウイルスの追加パラメータです．

利用しているスキャンワクチンと同じディレクトリに **VAKZIN.APP** という名前の追加パラメータファイルを作成して利用してください．

```
NAME  Jimi(NASITA)
DATA  02FH, 8AH 27H 80H F4H FFH 88H 27H 43H E2H F6H
DATA  039H, 6EH ECH FBH D2H FDH FFH 5CH ECH FBH
DATA  077H, 46H F9H FFH 47H FEH FCH 32H ECH
DATA  0F4H, 4BH FBH 32H E5H 7EH 05H EAH EEH 8AH
DATA  167H, 46H F1H FFH 49H FEH
IPL   T, 0
RAM   T, 200H
```

読者の声

<5月号の特集について>

■ 今月号から買い始めました．DOS/Vマシンの自作には非常に興味があり，適当な解説書を探していたので，正に渡りに舟でした．

記事の内容を参考にして，いつかは自作を試みてみようと思います．その際には，AT互換機ではなく，AT動かん機ができてしまわないことを祈るのみです．

▶**武下博英**

■ 創刊号から購読しています．創刊当時は他誌では読めない貴重な記事が多かったのに，最近内容がマンネリ化していると思う．今秋Windows95が発売されるので，NTや95上で動作するWin32プログラムの特集を他誌に先駆けて扱うなどの努力を編集部に期待します．

▶**かんちゃん**

■ ここのところ毎月買っています．今月号の特集は“だいたい知ってる”ことばかりでしたが，人に説明する必要にせまられたときのことを考えると“保存版”ですね．Windowsを始めて間がないので，最近の特集はとても助かっています．御不満の方もおられるでしょうが，この調子でいってくれればありがたい．

▶**ビン坊**

■ 今年の1月号から買っていますが，今回の特集はあまり必要でない情報が多かったと思います．

この特集のみでDOS/Vマシンをパーツから組み上げるのは難しいだろうし，かといって各ソフト類は“**SETUP.EXE**を実行”と書くほどのものとも思えません．ページ数などの都合もあるとは思いますが，もう少しつっこんだ内容にする，またはもっとさらっと流すのはいかがでしょうか？

▶**大薮祐司**

■ 特集をじっくり読ませてもらった結果，自分自身もメーカー製のDOS/Vマシンではなく，特集のような自作システムを手掛けてみたいものだと思いました．しかし，やはり秋葉原のような部品が入手しやすい店がないと無理かなとも思いました．名古屋のアメ横ではぜんぜんダメです．メーカー品ばかりです．この特集で取り上げた部品を，そのまま入手できる方法はないでしょうか？　また，次回もシステム立ち上げ部分の詳細を特集してほしいです．

▶**ヤマモト**

■ DOS/V作りは，大人のプラモデルといった感があり，今後もこのような企画を増々お願いいたします．貴誌は広告が少いのが良いのか悪いのか…．今後も，他誌とは一線をひいた内容をお願いいたします．

▶**286US**

■ 最近DOS/Vマシンを自作したので，特集につられて久しぶりに買いました．特集を読むとマシンの自作も奥が深いものだなと他人事のように思ってます．私の自作マシンが3時間でWindowsのインストールからSCSIのセットアップまで終えて，トラブルなく動作しているのはもしかしてまぐれなのかなー．ほとんど深く考えずに作ったんだけど．

▶**あきちゃん**

■ パソコンを作ってしまおうという「楽しみながら作るDOS/Vマシン100％活用法」は，トラコンならではの企画だった．小生の技術力では作ってみる気にはなかなかなれないが，DOS/Vマシンのハードが理解できたのは収穫だった．DOS/VマシンのハードはPC98より知られていない．低レベルでの各種デバイス(BIOS，シリアルポート，FDD，メモリ配置など)へのアクセス法の記事の充実を希望します．

▶**大内　定**

[編] I/Oレベルのハードウェア，BIOSのプログラミングについては，別冊「IBM PC&DOS/V活用ハンドブック」に詳しく解説してありますので，参考にしてください．

■ やけに軟弱なAT組み立ての特集ですね．ATの組み立てはおびえるほどキケンではありませんが，かといってカードを差せば動くというものでもないでしょう．また3/4はDOS/Windowsとからめていますが，PC UNIXやOS/2ユーザーにはたいして意味がありませんし，ある程度のレベルに達している人を対象にしなければ，非常に無責任な記事になってしまいますョ．「一発で組み立てが成功した」という想定での記事には無理がありますし，せめてトラブルシューティングの記事をのせてください．

▶**中村洋一**

■ 今月の特集はチョッとがっかりした．たしかに自分でパーツを組み立てて作るには，現在，資料が少ないのは実感してますが，せっかく，トラコンでやるんだから，もっと違ったアプローチがほしかった．

毎月，全部読みきるには2～3週間かかるのに，今月は3日でした．ザゼンのような苦しさが良いのに．

▶**PEET**

■ 特集は，PC/ATの組み立てについて体系的によくまとまっていると思います．最先端，最新のスペックを追わずに，実用的な組み合わせが解説してあったので，好感が持てました．PC/ATはいつのまにか部品が集まって，1台組み上げることができるようになってしまうので，ゼロから組み立てる人以外にも参考になると思います．

▶**三端子**

■ 先日，友人からマザーボード，I/Oカード，ミドル・タワーケース，ビデオカードをもらったので，残りの部品を買い足して自作のDOS/Vを作ろうとした矢先，今月号の特集にぶつかったため，さっそく購入して1頁目からナメるように読みました．私はハードウェアに関する知識がほとんどなかったので，大変役にたちました．

▶**篠田善規**

■ DOS/Vパソコン自作の真最中に何か参考になる

本はないかと本屋に行くと，トラコン5月号を偶然見つけて迷わず買ってしまいました．組み立て方の記事では多くの写真や図が載っていたので非常に参考になりました．今，私のオリジナルマシン(DX4/100 MHz)はすぐ側で元気良く動いております．　▶横田紀明

■ 本号は，パソコンを組み立てる面白さを良く紹介している．価格的にはメーカーブランドも自作より良いが，特殊マザーボードで拡張を制限している．長い目で見ると高い買物となる．私も世界トップブランドCOMPAQをリファレンスマシンとして所有してはいるが，テスト的な目的には自作AT互換機のほうが活躍している．　▶斎藤孝男

■ 楽しみながら作るDOS/Vマシン…の「楽しみながら作る」の部分は甘えているのではないか．もっとぴりっとした表現，特集のほうが良い．ただし，売り上げのことを考えるとどうなのだろう．　▶宮沢　仰

〈その他のご意見〉

■ Microcal社のOriginの購入を検討していたので，今月号はためになる記事「Originを使った波形シミュレーション」でした．「前編」となっているので，次号でもう終わりなのですか？ ぜひ連載のかたちにしてください．お願いします．　▶S.O

■ Sound BlasterのFM音源やPCM音源を直接I/Oから制御する記事を見たことがないので，ぜひ貴誌にやってもらいたい．同様のことを望むDOS/Vプログラマは多いはずです．　▶Big West

［編］現在，制作している別冊の中でSBのプログラミングについて取りあげる予定になっていますので，ご期待ください．

■ 特設記事Visual C++は参考になります．なおOLEサーバー，クライアントのプログラミングについて今一番知りたいのですが，良い参考書がありません．ぜひ記事にしていただきたいと思います．　▶人工生命

■ 最近MFCを使ってWindowsのDLLを書くようになった．それにしても**CString**はべんりだ．オブジェクトを確保してロックして使って，アンロックして，解放するなんてめんどうなことをしなくていい！　▶りょう

■「温故知新アルゴリズム考」は毎回楽しみにし，仕事にも何回か利用させていただきました．今回で最後なのが残念です．アルゴリズムについての記事は今後もあったほうが良いと思います．　▶正木幸一

■ 毎号楽しく読ませていただいています．連載の「温故知新アルゴリズム考」が終わってしまい，ちょっと

投稿募集のお願い

本誌では，読者の皆さんが作成されたプログラムなどを募集しています．もし，プログラムなどを発表したいと思われたなら，是非とも本誌編集部までご連絡ください．

お手紙でも，電話(03-5395-2124)でも，NIFTY-Serveを使ったメール(ID番号：HBG01366)でも，何でもOKです．

本誌掲載プログラムを利用される方へのお願い

本誌掲載プログラムにはすべて著作権があります．個人で利用される場合以外は著作権者の承諾が必要です．また，掲載プログラムを使用して生じたトラブル等については，CQ出版㈱ならびに著作権者は責任を負いかねますのでご了承ください．

CONGRATULATION

モニタープレゼント(5月号)当選者

① Microsoft Visual C++ Version 2.0
［マイクロソフト㈱提供］
千葉市・佐藤権一

② Microsoft Access for Windows Version 2.0
［マイクロソフト㈱提供］
明石市・松井利夫

③ ロータス プレミアム OFFICE2 for Windows
［ロータス㈱提供］
那覇市・佐原吉朗

④ 日本語Mathcad +5.0 Windows版
［住友金属工業㈱提供］
大阪市・赤路雄一

⑤ Turbo C++ 4.0J for DOS［ボーランド㈱提供］
高岡市・宮本　純

⑥ 一太郎 V6 for Windows［㈱ジャストシステム提供］
板橋区・北本秀夫

読者プレゼント(5月号)当選者

Ⓐ ラベル印刷ソフト カンタンプリント
揖保郡・福沢陽介

Ⓑ BTRIEVEのTシャツ
松山市・山田良作，上川郡・藤田雄一

CD-ROMプレゼント(5月号)当選者

★ Info-MAC V　各務原市・上田　誠
★ HyperStack　千歳市・小島秀隆　(敬称略)

訂正

● 1995年6月号

p.74 左↑8 「V.34が」を取る
p.85 〈図22〉(a)，(b)の図を入れ換える
p.126 本文，キャプションにおいて「クリティカルセッション」→「クリティカルセクション」

寂しい思いがしますね．ぜひまた，アルゴリズム関係の特集，連載をお願いしたいです．UNIX で動く Perl なんかを題材に選んでみてはいかがでしょうか？(最近の PC-UNIX 流行りにあわせて)　▶牧　和寛

▌Windows Programming FAQ に期待しています．できれば，C とクラスライブラリを分けてほしいです．Q：MFC で子ウィンドウ，ポップアップウィンドウを作る場合どのようにすればいいのですか？

ポップアップ用に派生クラスを作ってメインの **Create** の所で **new** しようとすると，派生クラスのコンストラクタが **Protected** なのでコンパイルが通りません(**CFnameWnd** のコンストラクタだとうまくいきますが…)．　▶斉藤正直

▌特集につられて初めて読ませていただきましたが，特集以外の記事もたいへん充実しており満足のいくものでした．特に「EXCEL 5.0 の VBA 活用マニュアル」は，現在携わっている業務に即役立つもので，今後の展開が楽しみです．特集に関しては，今回はごく初歩的なものにとどまっていると思いますので(それが一番大切なのですが…)，中級編，上級編を期待しています．　▶釜鈴　徹

▌C＋＋を理解している人の意識知識などを一つの集合とすると，「C＋＋を全く知らない意識」はその部分集合として存在しうるか？ それなら，この部分集合を対象に書かれた入門書はすべての人にわかりやすい入門書となりうる．ゆえに素人向けの素人にわかりやすい文を書くべきである．雑誌の入門記事を読んでいて思ったことです．「文を書くのはアマチュア」の人が多すぎる！　▶Monster P.

▌自分のパソコンがどんどんとオンボロになっていく．ソフトも GUI 化され，ハードディスクや増設メモリが完全に不足状態になってしまった．新しいものを買おうとすれば，何を選んでよいかわからなくなる．今だと P5-120 でハードディスク＝1 G バイト，メモリ 16 M バイト以上，CD-ROM，…と考えていると，…PCI 2.1 も出そうだし(^ ^;)　▶新屋

年間予約購読を受け付けています

トラ技コンピュータの年間予約購読ができます．ご希望の方は，CQ 出版社 営業部まで，または本誌のアンケートはがきを使ってお申し込みください．折り返し振り替え用紙をお送りします．購読料は送料，消費税込みで 1 年間10,320円です．

バックナンバのお知らせ

トラ技コンピュータのバックナンバが入手できます．ご希望の方は，お近くの書店あるいは CQ 出版社営業部(住所はディスク頒布の申し込み先と同じです)までお申し込みください．

モニタープレゼントのお知らせ

Information(p.149～p.153)で紹介した下記の製品を，メーカーのご好意によりモニタープレゼントいたします．本誌に添付の「リーダーズカード」に希望する製品名の番号をご記入のうえお申し込みください．締め切りは 1995 年 7 月 17 日(必着)です．当選者は 1995 年 9 月号で発表します．希望者多数の場合は，抽選させていただきます．

① まいと～く Ver.3［㈱インターソフト提供］(1 セット)

読者プレゼントのお知らせ

応募は，本誌に添付の「リーダーズカード」をご利用ください．締め切りは 1995 年 7 月 17 日(必着)です．当選者は 1995 年 9 月号で発表します．希望者多数の場合は，抽選させていただきます．

Ⓐ Compaq 社特製アナログ時計 (1 セット)
Ⓑ NetImage 社特製ポロシャツ (1 着)

ディスク頒布のお知らせ (今回は 1995 年 5 月号と 6 月号，7 月号の内容)

1995 年 7 月号の内容はニフティサーブの FDEVICE フォーラムにアップしてあります．

本誌 1995 年 7 月号で掲載したプログラムのソース・リストをフロッピで頒布します．ただし，市販の言語処理ソースなどにはすべて著作権がありますので一緒に頒布できません．コンパイラなどの処理系は読者自身がご用意ください．また，ソース・リストも読者の入力の手間を省くことが主な目的ですので，プログラム上のバグや本誌に掲載した内容以上の質問には応じられないことがありますのでご了承ください．同ディスクには 1995 年 5 月号および 1995 年 6 月号の掲載プログラム，トラコンInformation およびトラコン・データベースの最新版も入っています．ご希望の方は現金書留か定額小為替で代金 3,000 円(送料・税込み)と下記の申し込み書のコピー(手書き，ワープロでも可)を CQ 出版㈱デザインウェーブまでお送りください．なお，頒布メディアは MS-DOS フォーマットの 3.5/5 インチ 2HDです〔申し込み書にどちらかをチェックしてください．指定がない場合は 3.5インチ2HD(PC98 用)をお送りします〕．なお，今月号のディスク・サービスの受け付け期限は 1995 年 11 月末とします．

〈送り先〉

〒170
東京都豊島区巣鴨 1-14-2
CQ 出版㈱
デザイン・ウェーブ 行

◆申し込み書 TC9507　〈トラ技コンピュータ 1995 年 7 月号 ディスク・サービス〉

PC98	3.5☐	5☐
IBM	3.5☐	5☐

ご住所 〒　　－

お名前　　　　☎(　　)　　－

会員番号* CQ

＊資料請求で会員に登録されている方は会員番号をご記入ください．

コンピューティング・エッセイ 35

リアルワールドはどっちだ

北海道大学　山本　強

今年のゴールデンウィークも終わってしまい，たまった仕事をさばいていたら，突然地元の新聞社から電話で取材が入った．いつもは経済部からかかってくる電話が，なぜか今回に限って社会部，それも東京支社から．

何事かと思ったら，今世間を賑わしている某宗教団体をめぐる一連の事件とコンピュータネットワークの関わりを調べているとのことである．

確かに，今回の事件はコンピュータ，特にパソコンネットワークと関連が深いことも事実なのである．当の宗教団体が直接経営しているパソコンショップがあったり，自前で運営しているパソコンネットがあったりで，事件報道の早い時期から，そのハイテクぶりがかなりの話題になっていた．

その取材の中で記者の方が言うには，世界規模でみるとインターネットやパソコン通信で布教や信者の勧誘をする宗教団体はたくさんあるという．そして，“たかがパソコン通信”で，その人の人生を全く変えてしまうような宗教の布教や勧誘が出来るということも信じられなかったし，そんなことが現実にあることに驚いたという．

彼が取材した中には，パソコン通信だけでその宗教団体に接触し，いっさい教団関係者と物理的な接触なしに入信した人がいるのだそうである．

彼いわく，どういう心境で人はパソコン通信だけで全く知らない世界や団体を信じてしまうのか理解に苦しむとのことであった．

人それぞれなんだから良いのでは，と無責任な受け応えをしていたが，しだいに話がリアルワールドとバーチャルワールドということになって，私もちょっと真面目に考えはじめた．

バーチャルリアリティとかサイバースペースとか言われるコンピュータ上の仮想空間が話題になって久しいが，その対極にあるのが物理空間で生活する生身の人間，すなわちリアルワールドである．

私が生まれたときには家にコンピュータもなかったし，テレビが来たのも小学生になってからである．もともとリアルワールドしかない世界で生まれ育ってきた．

三つ子の魂百までではないが，リアルワールドつまり，食事をして体を動かす空間を“実”空間と考えて，仕事上つき合うコンピュータの中の仮想空間を“虚”空間と考えるようになっている．所詮，コンピュータの中の世界は仮の世界であるという認識である．

しかし，今大学生をやっている学生諸氏の世代の視点でみれば，小学校の低学年の時にすでにファミコンがあって，大学に入るとパソコン通信やインターネットが当たり前になっているのである．

ひょっとして，彼らの中にはコンピュータの向こうの世界を“実”の世界と考えていて，食事をしたり彼女とデートする生身の世界を“虚”の世界と考える人がいても不思議はないと思えてきた．

リアルワールドとバーチャルワールド，どっちが心地よいかと言えば，これはもうバーチャルワールドであろう．物理拘束が全ての行動を縛ってくれるリアルワールドでは，100 m はカールルイスをもってしても 9 秒は切れないし，社会のしがらみの中で発言すら制限されてしまう．

それと比べると，バーチャルワールドのなんと心地よいことか．この世界では実世界の物理拘束の時間，空間の拘束は存在しないし，社会のしがらみすら自分で消し去ることができてしまうのだから．

バーチャルワールドのただ一つの問題は，人間はその空間だけでは生きていけないということである．食事と排泄，そして子孫をつくること，これはバーチャルワールドでは出来ない．

したがって，バーチャルワールドを主たる空間として生きようと考えても，1 日何回かはリアルワールドに帰らなければいけない．

もっとも，どっちが実であっても，主たる世界をどちらにするかというだけの話であるから，所詮はライフスタイルの問題である．そんなことをとやかく言う気はまったくない．

そんなことを思って記憶をひもとくと，リアルワールドとバーチャルワールドがひっくり返っているのではないかと感じた人に会ったことがある．

まず，頂いた名刺に住所も所属もなく，名前とパソコン通信の ID だけが書いてあった人．きっと本職用の名刺は他にあるのだろうが，そっちは切れてもこっちは切らさないところに実と虚の逆転が感じられる．

また，ネットワーク上の言動から受けるイメージと実世界の発言から受けるイメージがまるで異なっている人も，この世界には多い．往々にして，ネットワーク上の性格の方がアクティブに見えるところから察するに，本当の自分はネットワーク上，つまり虚の世界に現れているのではないだろうか．

こういった新しい価値感は，ともすると異常な行動と取られがちである．しかし，異常と正常はどうやって区別するのだろうか．ある説によれば，異常と正常は，数の多い方を正常と定義し，少ない方を異常と定義するのだそうである．

だとすれば，今から 20 年後にバーチャルワールドを主たる世界と考える世代が多数となったあかつきには，我々リアルワールド型人類は異常となってしまうのである．異常と正常とは，かくもあやふやなものだと思う．

わかる 新しい技術力強化プログラム

エレクトロニクス・セミナー

新人セミナー
参加者募集中!!

セミナーの内容，申込み方法等についてはつぎのページ（出版案内）をご覧ください．



CQ出版社では，エレクトロニクス技術者に役立つ多くの実務書籍を発行していますが，昨年より読者の皆様のご要望に応えて，エレクトロニクスセミナーを開講しています．

4月～6月は，特に新入社員の方々の強い要望をいただいて，新人セミナーを開講中です．学校で一応の勉強はしてきたけれど，まだ電子回路にどうもなじめない方，技術系だけれどここで電子回路の常識をマスターしたい方などにおすすめします．またエレクトロニクスにはほとんど触れていなかったという方にも気軽に参加していただけるよう配慮してあります．

いま技術情報は市場にあふれていますが，その洪水の中でどんな情報を選ぶかが大切です．また最新情報にだけ流されて基本を忘れると，応用のきかない，狭い技術者になってしまいます．

このセミナーは『トランジスタ技術』に長い間執筆されているベテラン筆者の中から，またメーカの一流技術者の中から講師を選び，入門から実用技術までをわかりやすく説明します．

テキストも親切な図面や箇条書きの簡明な説明を多用し，見ているだけでも理解が進んでいく，ビジュアル・プレゼンテーション重点の方法です．あなたの実力養成のため，ご参加をお待ちしています．

マルチベンダ・システム構築技術・強化マガジン！

インターフェース増刊 OPEN DESIGN オープンデザイン 第9号 6月24日発売

連続特集 電子メールの導入・相互接続とグループウェア

B5判 160頁 定価1,800円 送料310円



パソコン電子メールの導入事例や，パソコン電子メールとインターネット電子メール・システムとの相互接続，そして，いま話題のグループウェアについては開発事例や活用事例に焦点をあてて特集します．

特集は，第1部「パソコン電子メールの導入事例」，第2部「電子メールの相互接続」，そして第3部「グループウェアへの発展」に分けられます．

第1部では，パソコン電子メールでトップシェアのcc:Mailなどを題材にして，パソコン電子メールの原理・機能・特徴から導入事例や活用事例を解説します．第2部では，cc:Mailとインターネット，およびcc:MailとMHSの相互接続，そして，インターネットとMHSの相互接続をとりあげます．第3部では，グループウェアの実現方法を，「電子メール拡張による方法」と「文書データベースによる方法」の二つに分けて，それぞれ実現技術と導入・活用事例を紹介します．

▶詳細解説　最新トランスポート・プロトコルXTP4.0

CQ出版社 〒170 東京都豊島区巣鴨1-14-2 営業部☎03-5395-2141 振替 00100-7-10665

徹底マスター 実戦力養成

わかるエレクトロニクス・セミナー

オリジナル・セミナー **第14回** フレッシュマンのための **「半導体パッケージングの基礎技術」**
（IC，LSI，ディスクリート半導体の組立技術入門）

開催日 1995年6月22日（木）～23日（金） 2日間 **参加費** 38,000円 **申込締切日** 6月19日
講 師 佐伯 準一氏（日立製作所 生産技術研究所 民生実装センター）

―内 容―

半導体素子のコストと信頼性はパッケージングで決まります．またパッケージングは材料，製造の多くのノウハウが必要で，素子の小型化とともに技術も急速に進歩しています．

本セミナーでは半導体パッケージングの基本技術を半導体メーカーの第一線技術者が入門からわかりやすく説明し，高速・多ピンの新パッケージにもふれます．半導体分野の新人技術者に必須のセミナーです．

①パッケージの形態と用途
②組立工程(1) ウェハーダイシング，ダイボンディング，ワイヤボンディング
③組立工程(2) プラスチックモールディング
④組立工程(3) リードの成形加工と表面処理，検査，包装
⑤パッケージの評価法と信頼性向上対策
⑥高速・多ピンパッケージの動向

著者セミナー **第15回** フレッシュマンのための **「CMOS ICの基礎とカスタムLSI設計への取り組み」**

開催日 1995年6月29日（木）～30日（金） 2日間 **参加費** 38,000円 **申込締切日** 6月27日
講 師 大幸 秀成氏（東芝 汎用LSI応用技術部 標準ロジック応用技術課）
副テキスト トランジスタ技術SPECIAL No.4「CMOS標準ロジックIC活用マニュアル」（定価1,540円）

―内 容―

電子技術の中核であるディジタル回路のハードウェアはCMOS標準ロジックがないと成立しません．CMOSは製造工程がやや長くなりますが，雑音許容度，低電圧動作，高速性，回路の簡単化などにすぐれた性能を持ち，ほとんどの超LSIはCMOSで作られています．またASICを含むカスタムLSIのプロセスもCMOS全盛です．

本セミナーは電子技術者として知らねばならないCMOS IC，LSIの構造，基本回路からカスタムLSIの設計にいたるまで，第一線のベテラン技術者がわかりやすく説明します．ディジタル技術をマスターしたい新人の方々に最適のセミナーです．

①MOSトランジスタとCMOS ICの構造
②CMOSによる標準ロジックIC
③CMOS基本ゲート回路の使い方
④CMOSフリップフロップの使い方
⑤その他のCMOSロジックICとLSI
⑥カスタムLSI設計への取り組み

オリジナル・セミナー **第16回** フレッシュマンのための **「ベーシック・ウエハープロセス」**
（第11回セミナーの追加）

開催日 1995年7月14日（木）～15日（金） 2日間 **参加費** 38,000円 **申込締切日** 7月7日
講 師 河野 博通氏（NEC システムASIC事業部 技術課長）
＊本セミナーは第11回の新人セミナーが定員超過のため締め切りとなりましたので，再度企画しました．

―内 容―

半導体チップを製造するためのウエハープロセスは，超微細化と自動化が進むなかで，基本工程が理解しにくくなっています．基本的な加工原理を知っておくことは半導体技術者にとって必須の条件です．

本セミナーは長年のウエハープロセスの経験をもつベテラン講師がプロセスのすべてを網羅し，基礎から最新の話題までを取りあげて，わかりやすく説明します．半導体若手技術者に最適のセミナーです．

①ウエハープロセス概論
②拡散，酸化プロセス
③イオン注入プロセスと電気的特性の管理
④リソグラフィ
⑤エッチングプロセス
⑥CVD，メタライズプロセス
⑦クリーン化技術
⑧LSIとプロセスフロー

CQ出版社

オリジナル・セミナー　第17回「電子回路の実装技術徹底マスター」

開催日　1995年7月27日(木)～28日(金)　2日間　**参加費**　38,000円　**申込締切日**　7月21日
講　師　盆子原　学氏（NEC　生産材料技術本部　実装技術開発部部長）

内　容

LSI，IC，半導体部品，受動電子部品の特性を充分に引き出し，設計した回路の機能を実現するためにはこれら部品の基板への実装技術が不可欠です．とくに最近要求される高速度，低雑音さらに大電力などの回路では実装技術そのものが回路特性を決定します．

本セミナーは電子回路の実装に関する基礎からノウハウまでを徹底的に理解するため，豊富な経験をもつ第一線のベテラン技術者が図面を多用してわかりやすく説明します．

①エレクトロニクスで重要な実装技術
②メインストリームのはんだ表面実装技術
③プリント基板設計のポイント
④部品実装装置の現状
⑤半導体用多ピンパッケージの実装技術
⑥新実装技術・半導体ベアチップ実装技術

今後のセミナー予定（詳しくは次号のこのページをご覧ください）

高周波通信技術の基礎	電源回路の基礎	作れば解るCPU
個別半導体素子の使い方	光素子の使い方	HDLによる論理回路設計入門

エルテスで力だめししてみませんか？

●エルテスとは…ELTES（Electronics Technology Evaluation System）はこのセミナーのために特別に開発された比較式エレクトロニクス技術力測定システムです．

●現在は試行期間ですが60問の4択式質問に40分で(レベル1の場合)答えていただき，六つの部門(回路基礎と部品，ICとLSI，アナログ回路，ディジタル回路，実装技術，コンピュータ）でのそれぞれの得点をいくつかの業界のトップ技術者の得点と比較します．ご自分がどの分野に強いのか，同業他社の技術者と比べてどうかを客観的に知る良いチャンスです．測定は強制ではありません．ご自分の測定結果は希望した場合は，ご自身および会社にお知らせします．

セミナーに参加ご希望の方は下の申込書に記入し，FAXで，あるいはコピーを右記まで郵送してください．

〒170 東京都豊島区巣鴨1-14-2
CQ出版株式会社内
エレクトロニクス・セミナー事務局
TEL:03-5395-2125　FAX:03-5395-1255

セミナー参加申込書

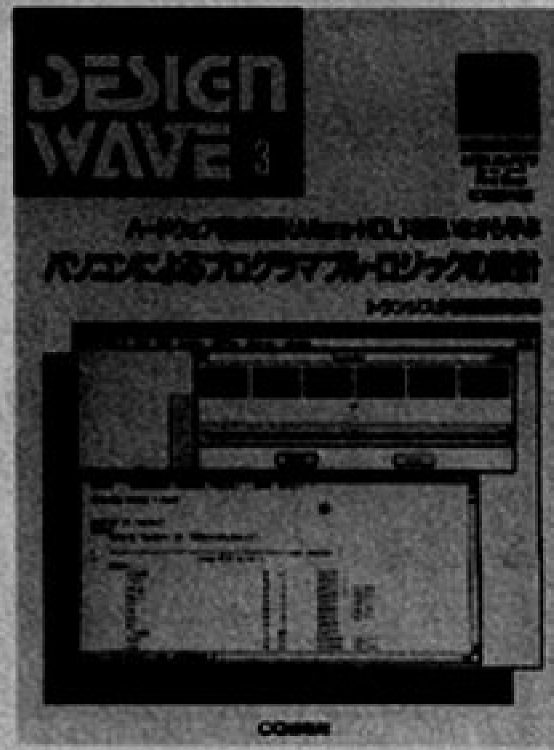
（申込書1枚に1名分をご記入ください）　（該当する□に✓をしてください）

参加者氏名			□新人セミナー全体の資料がほしい
会社・団体名			
所属・部門			
会社・団体所在地	〒		
電話番号		参加費請求先	□会社　□本人
FAX番号		送金の方法	□銀行振込　□現金書留
申込者(参加者と異なるとき)	氏名	所属	
参加希望セミナー回数	14□　15□　16□　17□		
エルテス(技術力測定)について	□参加してみたい	□上司に知らせてもよい	
得点を知らせてもよい上司の所属，お名前	所属，肩書	お名前	

デザインウェーブ・ブックス③　　好評発売中

ハードウェア記述言語(Altera-HDL)を使いながら学ぶ

パソコンによる プログラマブル・ロジックの設計



Windows 3.1対応のAHDLコンパイラ FirstStep(CQ版)付き！

B5変形判　256頁　3.5インチ2HD(1.44M)フロッピ×2枚付属

トランジスタ技術編集部編　定価3,200円(税込み)　送料380円

本書にはAHDL(アルテラ・ハードウェア記述言語)のコンパイラFirstStep(CQ版)が付属しており，AHDLによるロジック回路設計を使いながら学ぶことができます．FirstStep(CQ版)は，米国Altera社のプログラマブル・ロジック開発システムMAX＋PLUSII(Version 4.01)の機能を制限したもので，つぎの機能があります．

＊コンパイラ/7032(EPM7032専用)＊階層ディスプレイ＊メッセージ・プロセッサ＊テキスト・エディタ＊プログラマ(別途，専用ハードウェアが必要)＊変換ユーティリティ(ABEL→TDF，PALASM→TDF)

これらによって，AHDLで設計し，それをコンパイルし，MAX7000シリーズのプログラマブル・ロジックであるEPM7032デバイスにフィットさせることができます．ただし，デバイスに書き込むには，書き込み用ハードウェア(プログラマ)が必要です．

第１部　インストールおよび設計作業の概要

第２部　ＡＨＤＬマニュアル

第１章　はじめに／第２章　AHDLの使いかた／第３章　構文要素／第４章　設計の構成／第５章　文法／第６章　スタイル・ガイド／第７章　ほかの言語のユーザへの注意

第３部　実際の設計例

設計の手順／言語入力か論理式入力か／例１　バイナリ・カウンタ／例２　DRAMコントローラ

新刊常備店　上記の書籍をはじめ，CQ出版社の新刊を展示・販売しています

パルコブックセンター富貴堂	札幌市	011-214-2301
紀伊國屋書店札幌店	札幌市	011-231-2131
旭屋書店札幌店	札幌市	011-241-3007
金港堂本店	仙台市	022-225-6521
金港堂ブックセンター	仙台市	022-223-0979
高山書店	仙台市	022-263-1511
アイエ駅前店	仙台市	022-264-0718
加賀谷書店本店	秋田市	0188-32-3025
東北書店	郡山市	0249-32-0379
鹿島ブックセンター	いわき市	0246-28-2222
ヤマニ書房	いわき市	0246-23-3481
川又書店駅前店	水戸市	0292-31-0102
武石書店本店	勝田市	0292-73-1212
煥乎堂	前橋市	0272-23-1211
高崎戸田書店	高崎市	0273-63-5110
ナカムラヤ	太田市	0276-22-2001
須原屋本店	浦和市	048-822-5321
ブックセンター押田	大宮市	048-647-3141
芳林堂書店所沢店	所沢市	0429-25-5355
キディランド千葉店	千葉市	043-225-2011
スカイプラザ浅野書店	柏市	0471-64-2040
柏市新星堂カルチェ5柏店	柏市	0471-64-8551
三省堂書店神田本店	千代田区	03-3233-3312~5
書泉グランデ	千代田区	03-3295-0011
明正堂書店	千代田区	03-3257-0758
万世書房	千代田区	03-3255-0605
電波堂書店	千代田区	03-3255-8539
ラオックス・ザ・コンピュータ館	千代田区	03-5256-3111
書泉ブックタワー	千代田区	03-5296-0051
八重洲ブックセンター	中央区	03-3281-7760
東大生協本郷書籍部	文京区	03-3811-5407
三省堂書店新宿店	新宿区	03-3343-4871
紀伊國屋書店本店	新宿区	03-3354-0131
アクト・ブックス サンカマタ店	大田区	03-3735-1551
栄松堂蒲田店	大田区	03-3731-2241
大盛堂書店	渋谷区	03-3463-0511
紀伊國屋書店渋谷店	渋谷区	03-3463-3241
旭屋書店池袋店	豊島区	03-3986-0311
リブロ池袋店	豊島区	03-5992-6993
芳林堂書店池袋店	豊島区	03-3984-1101
くまざわ書店八王子店	八王子市	0426-25-1201
オリオン書店ルミネ店	立川市	0425-27-2311
パルコブックセンター吉祥寺	武蔵野市	0422-21-8122
久美堂小田急店	町田市	0427-23-7088
久美堂東急ハンズ店	町田市	0427-28-2772
有隣堂西口トーヨー店	横浜市	045-311-6265
有隣堂東口ルミネ店	横浜市	045-453-0811
有隣堂本店	横浜市	045-261-1231
文教堂書店新横浜駅店	横浜市	045-474-3535
有隣堂戸塚店	横浜市	045-881-2661
有隣堂BE店	川崎市	044-200-6831
文教堂書店小杉店	川崎市	044-711-0018
文教堂ブックセンター	川崎市	044-812-0063
住吉書房登戸店	川崎市	044-932-1000
有隣堂藤沢店	藤沢市	0466-26-1411
文教堂書店中央林間店	横浜市	0462-75-4165
有隣堂厚木店	厚木市	0462-23-4111
紀伊國屋書店新潟店	新潟市	025-241-5281
清明堂書店	富山市	0764-24-4166
うつのみや本店	金沢市	0762-34-8111
平安堂長野店	長野市	0262-26-4545
鶴林堂書店	松本市	0263-32-5340
自由書房	岐阜市	0582-65-4301
浜松谷島屋	浜松市	053-453-9121
マルサン書店宝塚店	沼津市	0559-63-0350
星野書店近鉄店	名古屋市	052-581-4796
三省堂書店名古屋店	名古屋市	052-562-0077
丸善書店名古屋店	名古屋市	052-261-2251
三洋堂書店上前津店	名古屋市	052-251-8334
精文館書店	豊橋市	0532-54-2345
シェトワ白揚	四日市市	0593-54-0171
駸々堂京宝店	京都市	075-223-1003
アバンティブックセンター	京都市	075-682-5031
駸々堂Version99	大阪市	06-353-4011
J&Pテクノランド	大阪市	06-634-1211
紀伊國屋書店梅田店	大阪市	06-372-5821
旭屋書店本店	大阪市	06-313-1191
オーム社書店大阪店	大阪市	06-345-0641
駸々堂書店心斎橋店	大阪市	06-251-0881
ヒバリヤ書店本社	東大阪市	06-722-1121
三和書店	尼崎市	06-413-1112
ジュンク堂三宮店	神戸市	078-392-1001
新興書房	姫路市	0792-85-3344
紀伊國屋書店岡山店	岡山市	086-232-3411
丸善シンフォニービル店	岡山市	086-233-4640
紀伊國屋書店広島店	広島市	082-225-3232
廣文館本通店	広島市	082-246-9581
宮脇書店	高松市	0878-51-3733
金高堂書店本店	高知市	0888-22-0161
ナガリ書店	北九州市	093-521-1044
井筒屋ブックセンター	北九州市	093-643-5520
紀伊國屋書店福岡店	福岡市	092-721-7755
紀伊國屋書店熊本店	熊本市	096-322-5531
リブロよしかわ	大分市	0975-56-6691

CQ出版社　〒170 東京都豊島区巣鴨1-14-2 営業部☎(03)5395-2141 振替 00100-7-10665

トランジスタ技術 スペシャル　好評発売中

〈年4回刊：今後の発売予定日 6/29，9/29，12/27〉

トランジスタ技術 SPECIAL No.50

エレクトロニクスの基礎と実用技術を濃縮したフィールド・ワーク・マガジン

B5判 176頁
定価1,800円 送料310円

特集 フレッシャーズのための電子工学講座

電磁気学の基礎から電子回路の設計，製作までをやさしく解説



「トランジスタ技術SPECIAL」も号を重ねること50冊目になりました．小誌創刊時は，ものを作る手段として，エレクトロニクスはあらゆる分野に利用されるようになってきた時代でした．創刊の大きな目標の一つに，「エレクトロニクス技術に関する実用技術の宝庫になりたい」という目論見がありました．近年，あまりにエレクトロニクスの進歩が速く，実際の製品に利用される技術と基礎的な技術との差が広がり，「なぜそう呼ばれているのか」「なぜそういうことをするのか」といった基本的な技術知識がとらえにくくなってきたように思います．No.50以降では，じっくり取り組めるように年4回刊として再出発し，基礎を見つめ直すため，本文中に出てくるテクニカル・タームや言葉足らずのところを補うようにしました．ご期待ください．

予告 No.51▶'95年6月29日発売
データ伝送技術基礎講座

トラ技コンピュータ 別冊　6月下旬発売予定

豊富な実例と的確な解説でC言語の基礎と応用力を養う！

基本C言語入門

桐山 清 著　B5判 280頁
予価2,400円 送料340円

本書では，Cプログラマに必要な基本知識を網羅しています．基本知識の解説には実例を提示し，実際にプログラムを動かしながら読んでいくことができるように構成されています．また，C言語を学ぶ人だけではなく，教える人にも役立つように基本知識の理解度を確認するための問題を各章末に掲載しています．C言語のテキストには，基礎事項を網羅しただけのものと応用プログラムの作り方だけのものという二つのカテゴリがありますが，本書はその両方をカバーしています．的確な解説でまとめたCの入門書としてもっとも優れた1冊です．

3.5インチ2DD FD付き



第1章：はじめに／第2章：データ型・式・文・型変換／第3章：演算子・優先順位と結合規則／第4章：コンソール入出力／第5章：プリプロセッサ／第6章：制御構造／第7章：関数・変数・通用範囲／第8章：配列／第9章：ポインタ変数／第10章：構造体・ビットフィールド／第11章：共用体・列挙型・ユーザ定義のデータ型／第12章：ファイル処理

CQ出版社　〒170 東京都豊島区巣鴨1-14-2 営業部☎(03)5395-2141 振替 00100-7-10665

CQ出版社から2つの新シリーズ

データ構造とアルゴリズム・シリーズ　好評発売中

コンピュータで問題を解決しようとするときのポイントが「データ構造とアルゴリズム」です．このポイントを6冊に分けて，グラフや図を多用し，難解になりがちな解説を直感的に理解できるように工夫した，待望の新シリーズです！　全巻ともC言語によるプログラム例を豊富に用意し，おもなソース・テキストを収録した3.5インチ・フロッピ・ディスク(2DD)を付けました．

① データ構造

浪平　博人 著
A5判　184頁
3.5″FD(2DD)付き
定価2,400円
送料310円



本書で解く「データ構造」は，問題を処理する際に，処理すべきデータをどのように表したらもっとも効率がいいかについて体系的に述べたものです．理科系だけでなく，調査・予測・分析などにたずさわる文科系の読者も強く意識して書かれています．

② アルゴリズム

浪平　博人 著
A5判　196頁
3.5″FD(2DD)付き
定価2,400円
送料310円



問題にあわせて表現した「データ構造」を，いかに効率的に処理するかについて問題解決の手順を体系化したものが，本書で述べる「アルゴリズム」です．生産計画や需要予測など，興味深い例題が随所に登場します．本書で情報処理の基礎力と応用力を確かなものにしてください．

つづいて，③ソート・検索　④情報表現技術　⑤再帰表現法　⑥問題解決事例も順次発売予定．乞うご期待！

新つくるシリーズ

エレクトロニクスのわかりやすい入門書が欲しいという声をよく耳にします．しかし，万人に対してわかりやすいというテーマを実現することは簡単ではありません．

「わかりやすい」ということを実現することはたいへんなのですが，エレクトロニクスについては「こうやって学べばよいのではないか」という答えがあります．それは，「自分の手で作ってみる」ということです．天才は閃きで物事を解明していくことができるかもしれませんが，凡人にとっては人真似から入るのも合理的です．

ということで用意したのが，この3冊の「新つくるシリーズ」です．いずれも『トランジスタ技術』誌，および『トラ技ORIGINAL』誌で掲載され，好評を博した記事のなかから，つくりたくなる記事，つくることを擬似体験できる記事をジャンルごとに再構成しました．真似をして体験することが最高の学習になると思いますが，読んでいるだけでも利用できそうなアイデアをふんだんにカバーしています．

No.1　〈好評発売中〉
つくるツール&測定器
おもな内容●ディジタル電圧計／ファンクション・ジェネレータ／カーブ・トレーサ／LCメータ／etc.

No.2　〈好評発売中〉
つくるオーディオ&ビデオ
おもな内容●オーディオ・アンプ／サウンド・プロセッサ／ビデオ・セレクタ／ビデオ・エフェクタ／etc.

No.3　〈6月24日発売予定〉
つくるオリジナル・グッズ
おもな内容●電子ゲーム／キッチン・タイマ／電子温度計／電磁波時計／ニカド電池充電器／紫外線メータ／etc.

●B5判●160頁●定価1,500円●送料310円

CQ出版社　〒170 東京都豊島区巣鴨1-14-2 営業部☎(03)5395-2141 振替00100-7-10665

別冊 インターフェース　　好評発売中



割り込みプログラミング技法

パソコンにおける割り込みメカニズムから デバイス・ドライバ作成まで

B5判 208頁 2色刷 3.5インチFD付き 定価2,400円(税込) 送料310円

本書では，国内のパソコンとして代表的な，PC/AT互換機(DOS/Vマシン)，PC-9801シリーズ，FM-TOWNSを取り上げ，これらの機種の，タイマ割り込み(8253，8254)，VSYNC割り込み，プリンタ割り込み，RS-232-C割り込み(8250B，8251A)に関して解説を行い，簡単なプログラムを作成します．パソコンで実際にプログラムを試しながら，ハードウェア割り込みを中心とした割り込みの活用法を探っていけるような構成です．また，通信用のデバイス・ドライバをC++によるオブジェクト指向プログラミングで作成した例や，DOSエクステンダ上の割り込みプログラミングの例も紹介します．

コンパイラの処理系としては，Visual C++，MS-C/C++，Borland C++，Turbo C++に対応しています．

■目次■

パソコンによるISDN通信

ISDN通信回線の利用と推奨版APIによるプログラミング　　6月下旬発売

多賀 直也〈財テレコム高度利用推進センター・監修〉 B5判 204頁 予価2,500円 送料310円

今やパソコンもネットワークに接続されるのがあたりまえの時代になりつつあります．ネットワークといえばLANを思い浮かべますが，LANの敷設されていない遠隔地とのデータ転送をしたいことも少なくありません．また，インターネットと結びたい人も少なくないでしょう．このとき，一般公衆回線＋モデムよりも高速伝送が可能な「ISDN公衆回線」を使用すれば，64kbpsでディジタル・データ伝送が行えます．すでに，ISDN公衆回線は，全国のほとんどの主要都市においてサービス運用がされており，その料金体系も，従来の電話用公衆回線とあまり変わりません．

本書は，パソコンによるISDN通信を行うためのハンドブックです．ISDNの基本を示したあと，ISDN通信に必要な機器やソフトを紹介しながら，ISDNを使用したシステム構築について，実例をあげながら解説しています．とくに，市販されているパソコン用ISDNソフトのほとんどは，FRAPI-AというAPI(ソフト・インターフェース)を使用していますが，このFRAPI-Aについてもくわしく説明しています．

■目次■

CQ出版社　〒170 東京都豊島区巣鴨1-14-2 営業部☎03-5395-2141 振替 00100-7-10665

＊1，3，4，8，12についてはフロッピ・ディスク版も6月上旬に発売を予定しています．



29年の実績……わが国唯一の半導体規格データベース

最新半導体規格表シリーズ　'95年版

●全20冊（A5判）．お得な全20冊のセット販売もあります（検索用3.5"FD付き）．
出版案内次ページの申込書をご利用ください．
★印のものは'95年版から定価が改訂になりました．

好評発売中

1 最新トランジスタ規格表　定価1,200円 国内で入手可能な製品のうち約6,000本のトランジスタを収録，型名，社名，用途，構造，最大定格（V_{CBO}，V_{CEO}，I_C，P_C），電気的特徴（I_{CBO}，h_{FE}，$V_{CE(sat)}$ $V_{BE(sat)}$ f_T，SW Time），コンプリメンタリ，電極接続，外形図等の情報のほかに，掲載データの中から電気的特性データをソートした索引を掲載しています．	**2 最新トランジスタ互換表**　定価1,200円 EIAJ登録の全トランジスタに対する主要メーカ8社（サンケン，三洋，東芝，日電，日立，松下，三菱，ローム）の互換可能なトランジスタを一覧表にしています．互換性の範囲は，最大定格，電気的特性，主要用途から見たもので，形状などの差異は無視しています．
3 最新ダイオード規格表　★定価1,300円 国内21社の約12,400品種のダイオードを収録（保守・廃品種を含む）．これらを，一般整流用，整流用アバランシェ，整流用ショットキ・バリア，整流用ダイオード・モジュール/スタック，小信号シリコン，シリコン・ポイント・コンタクト…などに21分類し，規格と外形図を掲載しています．	**4 最新FET規格表**　★定価1,300円 国内主要10社のEIAJ登録，各社名称のFETを収録，最大定格，電気的特性のほか，外形図，コンプリメンタリ，用途，構造，チャネル，モード等を掲載しています．また，海外主要8社のパワーMOS FETについても，最大定格，電気特性の中から主要項目をピックアップしています．
5 最新電力用素子規格表　★定価1,300円 国内主要メーカ十数社のパワー・デバイス（サイリスタ／バリスタ・ダイオード／UJT／各種モジュールの最大定格，電気的特性，外形等をコンパクトにまとめています．電気的特性は，電力制御回路設計に必要なものを中心に掲載しています．	**6 最新光表示素子規格表**　定価1,200円 国内外のLEDランプ，LEDディスプレイなど，表示用として汎用性の高い光半導体素子の電気・光学的特性，最大定格等を掲載しています．外形図が載っているので本格的データが活用できます．
7 最新高周波デバイス規格表　★定価1,300円 高周波ダイオード，トランジスタ，FET，パワー・モジュール，RF/IFアンプ・モジュールや関連のICの最大定格，電気的特性，ピン接続図を掲載しています．ほかに，保守・廃品種一覧，用途別早見表も各分野別にまとめています．	**8 最新74シリーズIC規格表**　定価1,200円 TTL IC，HCMOS，Advanced CMOS，Bi-CMOSのピン接続図，主要特性，パッケージ情報，用途別分類表を収録しています．また，まだ使う人が少ないものなどをまとめた，準推奨品一覧も掲載しています．3Vデバイスも追加されています．
9 最新CMOSデバイス規格表　定価1,200円 CMOS ICファミリで広く用いられているRCA社，モトローラ社オリジナルの4000B/4500B/5000/22000シリーズ，アナログ・スイッチ，スペシャル・ファンクションICについて，シリーズ別，番号順に，特徴，ピン接続図，ロジック・ダイヤグラム，スイッチング特性，他社相当品などを掲載しています．	**10 最新RAM規格表**　定価1,200円 Static RAM，Dynamic RAM，画像用RAM，デュアル・ポートRAMをビット・サイズ順に並べて，ピン接続，特徴，動作表，ブロック図，スイッチング特性，パッケージ情報を掲載しています．
11 最新PLD/ROM規格表　定価1,200円 PLD，UV-EPROM，EEPROM，PROM，One Time-EPROM，Flash-PROMについて，ピン接続，動作表，ブロック図，波形，スイッチング特性，電源，入力，出力／測定電流，パッケージ情報を掲載しています．	**12 最新モノリシックOPアンプ規格表**　★定価1,300円 モノリシックOPアンプICの特徴，ピン接続図，標準回路，電気的特性（入力オフセット電圧／電流，入力バイアス電流，入力抵抗，入力電圧，電圧利得，出力電圧，出力インピーダンス，同相信号除去比，電源電圧変動除去比，電源電圧／電流，スルーレート等），セカンドソース，類似品などを1頁単位にまとめています．
13 最新光インターフェース素子規格表　定価1,200円 赤外LED，レーザ・ダイオード，フォト・ダイオード，フォト・トランジスタ，フォトIC，フォト・カプラ，光通信モジュールについて，最大定格，電気・光学的特性を一覧表にまとめ，ピン接続と外形図も掲載しています．	**14 最新リニアIC規格表（アナログ機能IC編）**　★定価1,300円 産業用リニアICのうち，アナログ機能のものを収録しています．タイマ，VCO，関数発生器，PLL，基準電圧IC，コンパレータ，アナログ・スイッチ，*V-F*，*F-V*コンバータ，その他に分類し，特徴，同等品，ピン配置，最大定格，電気的特性，等価回路等を掲載しています．
15 最新リニアIC規格表（増幅用IC編）　定価1,200円 産業用リニアICのうち増幅用として使われるICを収録しています．低周波電圧増幅器，低周波電力増幅器，差動計測アンプ，広帯域アンプ，ゲイン可変アンプ，ゲート制御アンプ，その他のアンプに分類しています．特徴，パッケージの種類，ブロック図または回路図，最大定格，電気的特性を掲載しています．	**16 最新電源用IC規格表**　★定価1,300円 3端子レギュレータ（固定出力，可変出力），低ドロップアウト型3端子／スイッチング・レギュレータ，CMOS3端子／シリーズ・レギュレータ等を収録，特徴，最大定格，電気的特性，ピン配置，ブロック図，回路例等を掲載しています．
17 最新インターフェース素子規格表　定価1,200円 RS-232Cをはじめとするライン・ドライバ／レシーバ，ライン・トランシーバ，汎用のパワー・ドライバであるペリフェラル・ドライバ，トランジスタ・アレイなどを一覧表にまとめ，主要製品の最大定格，電気的特性，ピン配置，ブロック図などを掲載しています．	**18 最新A-D/D-Aコンバータ規格表**　定価1,200円 A-D/D-Aコンバータ，サンプル&ホールド，マルチプレクサ／4ビット～16ビットまで各種方式別に掲載しています．回路設計に最低限必要と思われる特性，ピン接続，機能ブロック図，特徴，基本回路例などを掲載しています．
19 最新マイコン周辺LSI規格表　定価1,200円 マイコン周辺LSIの代表的品種約200品種を，各2頁にまとめて掲載しています．割り込み，パラレル，シリアル，グラフィック，クロック，タイマ，DMA，LANなど20数種に分類し，ピン接続とその機能を収録しています．	**20 最新ビデオ信号処理用IC規格表**　定価1,200円 ビデオ信号処理に必要な各種デバイスを広範囲に収録しました．映像・色・輝度信号処理，同期信号処理，オンスクリーン・ディスプレイ，インターフェース，アンプ，スイッチ，A-D/D-Aコンバータ，ディジタル信号処理，メモリ，その他多数のICを掲載しています．

CQ出版社　〒170 東京都豊島区巣鴨1-14-2　営業部☎03-5395-2141　振替 00100-7-10665

デザインウエーブ・シリーズ　新世代エレクトロニクス・エンジニアのための設計ツール・ソフトウェア

DESIGN WAVE

3.5インチ フロッピ・ディスク2枚

好評発売中　定価14,610円　送料390円

(PSpiceはMicroSim Corporationの登録商標です)

回路図エディタ+電子回路シミュレータ

PSpice®/CQ版スーパーパック



PSpice®/CQ版スーパーパックは電子回路シミュレータPSpice/CQ版 Ver.5に専用の回路図エディタを付けたパッケージです．はじめてアナログ・シミュレータを使おうとする人に向けて発売/

〈おもな特徴〉

- 付属の回路図エディタ（PSEDITOR）で回路図をかけば，自動的にPSpice®用のネット・リストに変換可能.
- “シェル”による対話形式でわかりやすい操作.
- OPアンプ，トランジスタなど実用的なパーツ約30種類をパーツ・ライブラリ内に収録.

〈動作環境〉

▶対応機種：PC9801シリーズまたは互換機　▶ハード・ディスク：4Mバイト以上
▶MS-DOS：3.0以降　▶マウス：必要
▶メイン・メモリ：640Kバイト以上　(解析できる回路のノード数は64，変数は100未満)

※解析に使用するサーキット・ファイルなどは小社から発売中の PSpice/CQ版(Ver.5)と互換性があります.

'95年版最新半導体規格表

全20冊セット販売　全規格表用「検索ソフト」が付いて(3.5インチ)
定価24,700円(送料サービス)

好評発売中！

＊数に限りがございますのでお早めにお申し込みください．

ご希望の方は下記のいずれかの方法でお申し込みください．

①**現金書留**（下記の申込書利用）
宛先：〒170 東京都豊島区巣鴨1-14-2
ＣＱ出版㈱ 営業部 '95規格表セット販売係

②**郵便振替**（郵便局の振替用紙に「'95規格表セット」と明記してください）
口座番号：00100-7-10665

③**書店へお申し込み**（下記の申込書にご記入のうえ，書店へご持参ください）

------------ キ リ ト リ 線 ------------

書店使用欄
書店印

'95年度版「最新半導体規格表」セット販売申し込み書

ご住所：〒	会社名：
お名前：	所属部課名：
☎　（　　）　－	☎（　　）　－

▼送付先の住所と宛名を記入してください（送付ラベル用）

〒

様

取引取次店

トーハン　日販
大阪屋　栗田
中央社　協和
太洋社　誠光堂

＊太枠内のみご記入ください．

CQ出版社　〒170 東京都豊島区巣鴨1-14-2　営業部☎03-5395-2141　振替00100-7-10665

好評発売中

マルチベンダ・システム構築技術・強化マガジン

B5判
定価1,800円 送料310円

インターフェース増刊
オープンデザイン
OPEN DESIGN

第8号 好評発売中

全面特集 電子メール・システム完全マスタ



インターネットがらみで電子メールが注目されていますが，例によって，本誌は，電子メールの仕組みや実現法などの技術にしぼった特集です．題材は，RFCで情報が公開されているUNIX/インターネット・メールの世界が中心．これらをパソコン側からどう使うかといった話題にもふれます．

▶特集内容

電子メール入門／UNIX電子メールの使い方／UNIX電子メールのシステム管理（sendmailの設定法）／電子メールの動作メカニズム（SMTP/UUCP/POP/メール・ゲートウェイ）／パソコンからインターネット・メールにアクセスする／インターネット・メールのマルチメディア拡張MIME解説／MAPI・VIMとWindowsアプリへのメール機能組み込み法　ほか

▶詳細解説　Windows NT 3.5 のネットワーク環境

No.1 集中特集 SCSI完璧リファレンス

▶創刊1号は，PCやWSをはじめスーパコンピュータにまで装備されている外部機器接続のための標準規格SCSIの大々的な特集です．①SCSI装置の動作とコマンド理解，②PCやWSの機種別SCSI研究とデバイス・ドライバ，③トラブル事例とその対策，などをポイントにした172頁の集中特集です．緊急企画としてPCMCIA(PCカード)詳細解説も取り上げています．

No.2 総力特集 Macintoshインターフェース&ネットワーキング

▶Macのインターフェースとネットワーキング技術を特集します．①シリアル・ポート/SCSIポート/ADBポート/NuBus & PDSポートと，各ポート対応マネージャの詳細　②AppleTalkによる小規模LANから，WSや大型ホスト混在のオープン・システム事例やWANへの展開まで．創刊号から連続のPCMCIA，2号はソフトウェア編．

No.3 集中特集 イーサネットとTCP/IP

▶イーサネット・カード，ドライバ，TCP/IPプロトコル・スタック，TCP/IPのAPIとアプリケーションについてまとめます．▶サーバとしてのUNIXマシンを調査したあと，LANの高速化や広域化，それにイーサネットの負荷率とエラー率，ネットワーク管理などについてながめていきます．

No.4 全面特集 UNIXネットワーキング実践編―インターネットのために

▶「LANの種類と接続」や「モデムを介した遠隔接続」について整理します．▶「TCP/IPとUUCPのソフト設定法」につづいて，「電子メールと電子ニュースの設定と利用法」を解説します．▶パソコンとUNIX接続法やネットワークの規模拡大，それにセキュリティ技術にもふれます．▶連続企画のPCMCIAはVer2.1最新情報です．

No.5 総力特集 ネットワーク・ファイルシステムの研究―NFS/NIS/RPC

▶ネットワーク・ファイル・システムの構造について調べ，NFSの利用法やPC-NFSについて整理します．▶NIS(ネットワーク設定情報の集中管理システム)の使い方と応用例，RPC(遠隔手続き呼び出し)の原理とプログラミングを研究し，NFSプロトコルとその実装について集中解説します．

No.6 集中特集 インターネットワークのトラブル対策

▶ネットワーク・アナライザの種類と活用法▶物理層/データリンク層，TCP/IPのあらましとトラブル対策▶トークンリング/SNAとプロトコル解析の実際▶ISDNによるLAN-WAN接続とトラブル対策▶NetWareとマルチプロトコルなどの事例を紹介します．

No.7 全力特集 PCIバスの詳細と応用へのステップ

▶　特集は，PCIの概要からはじまって，アービトレーションやインタラプト，それにコンフィギュレーションや64ビット拡張などについて丁寧に解説しています．PCIは新しいバスであるため，構成が非常にすっきりしており，マイコン・バス入門としても読める，わかりやすい特集に仕上がりました．

CQ出版社　〒170 東京都豊島区巣鴨1-14-2 営業部☎(03)5395-2141 振替 00100-7-10665

デザインウェーブ・シリーズ/半導体規格表FD版/トラ技スーパーキット お申し込みのご案内

下記の商品は郵便振替か現金書留でお買い求めいただけます.

■ **郵便振替**　郵便局の振替用紙をご利用ください. 通信欄にご希望の商品名を明記し, 代金の合計をお振込みください.

●**口座番号　00100-7-10665**

■ **現金書留**　下記の申込書に必要事項をご記入のうえ, 代金の合計とともにお送りください.

●**宛先：〒170 東京都豊島区巣鴨1-14-2**
ＣＱ出版㈱

★なお, 商品の発送には入金確認後１～２週間程かかりますので, あらかじめご了承ください.

■ 申 込 書 ■

商品名	定価(送料)	下欄に希望数量を書き込んでください
① 回路図エディタ＋電子回路シミュレータ PSpice®/CQ版スーパーパック (PC9801用 3.5インチのみ)	14,610円 (390円)	
② パソコン用電子回路シミュレータPSpice/CQ版 (Ver.5) PC9801用 (3.5インチ or 5インチ)	11,640円 (390円)	3.5″ ／ 5″
③ パソコン用電子回路シミュレータPSpice/CQ版 (Ver.5) PC/AT, J-3100用 (3.5インチのみ)	14,640円 (390円)	
④ パソコン用回路図エディタTango-SCH/CQ版 —'95年度版— (3.5インチのみ)	9,610円 (390円)	
⑤ 半導体規格表ディスク版 トランジスタ・データ・ライブラリ —'95年度版— (3.5インチのみ)	9,610円 (390円)	
⑥ 半導体規格表ディスク版 ダイオード・データ・ライブラリ —'95年度版— (3.5インチのみ)	9,610円 (390円)	
⑦ 半導体規格表ディスク版 FETデータ・ライブラリ —'95年度版— (3.5インチのみ)	9,610円 (390円)	
⑧ 半導体規格表ディスク版 74シリーズICデータ・ライブラリ —'95年度版— (3.5インチのみ)	9,610円 (390円)	
⑨ 半導体規格表ディスク版 OPアンプ・データ・ライブラリ —'95年度版— (3.5インチのみ)	9,610円 (390円)	
⑩ 半導体規格表総索引 (FD版) (3.5インチ or 5インチ)	6,000円 (送料込み)	3.5″ ／ 5″
⑪ トラ技スーパーキット DAIレシーバ (「トランジスタ技術」'92年8, 9月号参照)	20,000円 (送料込み)	
⑫ トラ技スーパーキット 実験用可変電源 (「トランジスタ技術」'92年10, 11月号参照)	20,000円 (送料込み)	
⑬ トラ技スーパーキット ディジタル・マルチメータ (3.5インチ or 5インチ) (「トランジスタ技術」'93年3, 4, 8, 9月号参照)	20,000円 (送料込み)	3.5″ ／ 5″

送付先

＊送付先が会社の場合は部課名も明記してください.

ご住所：〒

お名前：　　　　☎(　　　　)　　　—

■ ②④⑩⑬の商品につきましては, ご希望のメディアの欄に数量を明示してください. 指定のないときは3.5″をお送りいたします.

DESIGN WAVE

デザインウェーブ・シリーズ直接販売商品

FLEXデザイン・キット/CQ版
PARTHENON/CQ版

ご購入方法のご案内

デザインウェーブ・シリーズの「FLEXデザイン・キット/CQ版」および「PARTHENON/CQ版」は，直接販売商品のため，ご購入の際は，下記のいずれかの方法でお求めください．

なお，一部の書店でも展示販売しております．詳細は営業部(☎03-5395-2141)まで．

ご購入の方法

(1) 郵便振替による方法

郵便局の振替用紙をご利用ください．

振替用紙の「通信欄」に下記の「申し込み書」に準じて，商品名，数量をご記入のうえ，代金の合計をお振り込みください．

- **加入者名　ＣＱ出版株式会社**
- **振替口座番号　00100-7-10665**

(2) 現金書留による方法

下記の「申し込み書」(コピー可)に必要事項をご記入のうえ，代金の合計とともに現金書留にて，小社営業部までお送りください．

- **宛先：〒170　東京都豊島区巣鴨1-14-2**
  ＣＱ出版株式会社　営業部

(3) ブックサービスをご利用いただく方法

右記ブックサービス(株)へお申し込みいただくと，商品をお届けしたときに，配達料(数量，代金にかかわらず380円)を添えてお支払いいただくことで，商品をご購入いただけます．なお，お支払いにはクレジット・カードもご利用になれます．

- **申し込み方法**

下記の「申し込み書」に準じて，出版社名(ＣＱ出版社)，商品名，数量とともに，送付先ご住所，会社名，部課名，お名前，電話番号をブックサービス㈱まで直接ご連絡ください．その際，葉書，電話またはFAXのいずれかの方法でお願いいたします．

- **申し込み先**

〒113　東京都文京区本郷1-28-10　本郷TKビル1階
ブックサービス株式会社
TEL (03) 3817-0711　FAX (03) 3818-5969

- **ご利用可能なクレジット・カード**

JCB, VISA, DC, UC, ミリオン, 共同

★ 郵便振替および現金書留による「直接販売商品」のご購入につきましては，送料は小社負担とさせていただきます．なお，「直接販売商品」の発送には，入金後1～2週間かかることがございますので，あらかじめご了承くださるようお願い申しあげます．

CQ出版社直接販売商品　申し込み書

直接販売商品　商品名	定価(税込み)	数量
FLEXデザイン・キット/CQ版	49,000円	個
PARTHENON/CQ版	35,000円	個

送付先	
	ご住所：〒
	会社名：　　部課名：
	お名前：　　☎ (　　)　　―

常備書店一覧

下記の書店では，CQ出版社の商品を多数取り揃えております．また，●印該当書店では分野ごとの商品を常備しておりますのでご利用ください．なお，新刊書籍については出版案内4ページの新刊常設協力店で取り扱っておりますので、あわせてご利用ください．

書店名	電話	常	規	D	P	デ
北海道						
紀伊國屋書店札幌店	011-231-2131	●	●		●	
旭屋書店札幌店	011-241-3007	●	●	●	●	●
パルコブックセンター富貴堂	011-214-2301	●	●			
丸善南一条店	011-241-7251		●			
ダイヤ書房本店	011-712-2541	●				
[illegible]	011-890-2586	●				
旭川冨貴堂本店	0166-26-6100		●			
森文化堂	0138-23-3238	●				
丸徳小田原書店	0154-31-2870	●				
ザ・本屋さん柏林台店	0155-33-5020	●				
明屋書店中標津店	01537-2-8451	●				
青森県						
紀伊國屋書店弘前店	0172-36-4511	●	●			
岩手県						
東山堂書店	0196-23-7121	●				
東山堂ブックセンター	0196-53-6464	●	●			
さわや書店	0196-53-4411	●				
伊吉書院三日町店	0178-44-1917	●				
小田島書店	0197-63-3351	●				
宮城県						
金港堂	022-225-6521	●	●	●	●	●
金港堂ブックセンター	022-223-0979	●	●			
丸善仙台一番町店	022-224-8018	●				
アイエ駅前店	022-264-0718	●				
高山書店	022-263-1511	●	●			
東北大学生協工学部書籍部	022-223-4518	●	●			
ブックスなにわ仙台中野栄店	022-258-3133	●				
ブックスなにわ塩釜店	022-363-3603	●				
秋田県						
加賀谷書店本店	0188-32-3025	●				
山形県						
八文字屋	0236-22-2510	●	●			
八文字屋エビスヤ店	0235-24-3022	●	●			
福島県						
東北書店	0249-32-0379	●	●			
ヤマニ書房本店	0246-23-3481	●	●			
鹿島ブックセンター	0246-28-2222	●				
茨城県						
ツルヤブックセンター水戸駅前店	0292-21-3223	●				
川又書店本店	0292-24-2047		●			
川又書店駅前店	0292-31-0102	●	●			
大野書店	02928-2-2098		●			
弘文堂書店	0294-24-2233	●	●			
田所書店	0294-22-5537	●				
武石書店本店	0292-73-1212	●				
マルエス神栖店	0299-92-1233	●				
ブック&レコードやまうち	0298-31-5756	●				
栃木県						
宇都宮ブックセンター	0286-27-7700	●	●			
落合書店オリオン店	0286-34-3777	●				
落合書店東武ブックセンター	0286-34-8274		●			
八重洲ブックセンター[illegible]	0286-27-8588	●	●			
イズミヤブックセンター小山店	0285-21-1230	●				
進駸堂駅ビル店	0285-25-1522	●				
進駸堂城南店	0285-28-2288	●	●			
宮脇書店小山店	0285-28-6555	●				
群馬県						
煥乎堂	0272-23-1211	●	●			
戸田書店日吉店	0272-32-7300	●				
高崎戸田書店	0273-63-5110	●	●			
[illegible]	0273-27-3961	●				
白木屋書店	0277-45-2175	●				
藤岡戸田書店	0274-22-2469	●	●			
ナカムラヤ	0276-22-2001	●	●			
埼玉県						
押田謙文堂	048-641-3141	●				
ブックセンター押田	048-647-3141	●				
新栄堂書店大宮店	0486-44-2345		●			
須原屋本店	048-822-5321	●				
須原屋コルソ店	048-824-5321	●				
岩渕書店川口店	0482-52-2190	●	●			
時習堂	0494-22-1510	●				
パルコブックセンター新所沢店	0429-98-8175	●				
芳林堂書店所沢店	0429-25-5355	●				
戸田書店本庄店	0495-24-2160	●				
文教堂書店春日部店	048-752-7666	●				
須原屋蕨店	048-444-1211	●				
文教堂書店新座店	048-479-7070	●				
黒田書店川越店	0492-25-3138		●			
黒田書店霞ケ丘店	0492-61-0513		●			
田中書房	0485-81-1619		●			
千葉県						
キディランド千葉店	0472-25-2011	●				
そごうブックセンター	043-245-8331	●				
多田屋セントラルプラザ店	043-224-1333	●				
中島書店	043-224-2448	●				
旭屋書店船橋店	0474-24-7331	●				
芳林堂津田沼店	0474-78-3737	●				
[illegible]	0474-77-2611		●			
BOOKS松田	0438-23-4210	●				
新星堂カルチェ5柏店	0471-64-8551	●	●			
スカイプラザ浅野書店	0471-64-2040	●				
博文堂書店君津店	0439-55-6657	●				
東京都						
三省堂書店神田本店	03-3233-3312	●	●	●	●	●
書泉グランデ	03-3295-0011	●	●	●	●	●
東京堂書店	03-3291-5181	●				
巌翠堂	03-3291-1362	●				
明正堂秋葉原店	03-3257-0758	●	●			
万世書房	03-3255-0605	●	●	●	●	●
電波堂書店	03-3255-8539	●	●	●	●	●
書泉ブックタワー	03-5296-0051	●	●	●	●	●
ラオックス ザ・コンピュータ館	03-5256-3111	●	●	●	●	●
T・ZONEミナミ店	03-3257-1025	●	●			
八重洲ブックセンター	03-3281-1811	●	●			
丸善日本橋店	03-3272-7211	●				
丸善ブックネット	03-3273-1041		●			
東大本郷生協書籍部	03-3811-5407	●	●		●	
紀伊國屋書店本店	03-3354-0131	●	●			
西武新宿ブックセンター	03-3208-0380	●				
三省堂書店新宿西口店	03-3343-4871	●				
早大理工生協	03-3200-6083	●	●			
芳林堂書店高田馬場店	03-3208-0241	●				
新栄堂書店亀戸駅ビル店	03-3638-2345	●				
虎ノ門書房田町店	03-3454-2571	●				
ソーブン堂書店武蔵小山店	03-3783-5801	●				
ACTサンカマタ店	03-3735-1551	●	●			
KADOYA ACT4	03-3733-7511	●				
栄松堂蒲田店	03-3731-2241	●				
武蔵工業大学書籍部	03-3730-6340	●				
三省堂書店渋谷店	03-3407-4545	●				
大盛堂書店	03-3463-0511	●	●	●	●	●
旭屋書店渋谷店	03-3476-3971	●	●			
紀伊國屋書店渋谷店	03-3463-3241	●	●			
文教堂書店代々木上原店	03-5454-1270	●				
書原杉並店	03-3313-4778	●				
ブックセラーズ西荻	03-3395-6566	●				
新栄堂書店サンシャインシティアルパ店	03-3988-0181	●				
新栄堂書店	03-3948-2345		●			
旭屋書店池袋店	03-3986-0311	●	●	●	●	●
池袋芳林堂書店	03-3984-1101	●	●			
リブロ池袋店	03-5992-8800	●	●			
文教堂書店光ヶ丘店	03-5372-0864	●				
くまざわ書店八王子店	0426-25-1201	●	●			
ブックワールド棚生堂本店	0426-22-8261	●				
有隣堂八王子書籍部	0426-35-5060	●				
オリオン書店ルミネ店	0425-27-2311	●	●			
新星堂フロム中武店	0425-25-9892	●	●			
三成堂書店国分寺店	0423-25-3211	●	●			
パルコ吉祥寺店	0422-21-8122	●	●			
紀伊國屋書店吉祥寺東急店	0422-21-5543	●				
真光書店	0424-87-2222	●				
久美堂本店	0427-25-1330	●	●			
久美堂小田急町田店	0427-27-1111	●	●			
久美堂東急ハンズ店	0427-28-2772	●	●			
有隣堂町田東急店	0427-23-3018	●				
文教堂書店小金井店	0423-86-0161	●				
文教堂書店新小金店	0423-82-8885	●				
文教堂書店小平店	0423-43-9229	●				
文教堂書店新小平店	0423-46-1511		●			
戸田書店日野店	0425-85-2260	●				
ブックスタマ東村山店	0423-92-4264	●				
三石堂書店	0423-21-0969	●				
東西書店	0425-75-5061	●				
ブックス・タマ福生店	0425-52-8141	●				
くまざわ書店桜ケ丘店	0423-37-2531	●				
住吉書房稲城長沼店	0423-78-5266	●				
いけだ書店高輪店	03-3441-8347	●				
神奈川県						
有隣堂本店	045-261-1231	●	●			
有隣堂横浜西口トーヨー店	045-311-6265	●	●		●	●
有隣堂横浜東口ルミネ店	045-453-0811	●	●			
有隣堂ランドマーク店	045-222-5500	●				
横浜そごうブックセンター	045-465-2111	●				
住吉書房新横浜店	045-402-6401	●				
文教堂書店新横浜駅前店	045-474-3535	●	●			
有隣堂戸塚店	045-881-2661	●	●			
有隣堂たまプラーザ店	045-903-2191	●				
文教堂書店たちばな店	045-962-3560		●			
文教堂書店中山店	045-937-0028		●			
文教堂書店港北ニュータウン店	045-941-6540		●			
文教堂書店栄上郷店	045-895-3220	●				
綱島みどり	045-541-8228		●			
文学堂	044-244-1253	●	●			
有隣堂川崎BE店	044-200-6831	●	●			
北野書店	044-511-5491	●				
文教堂書店小杉店	044-711-0018	●				
文教堂書店溝ノ口店	044-811-8258	●				
文教堂書店ブックセンター	044-812-0063	●	●	●	●	●
流水書房KSP店	044-819-2313	●	●			
住吉書房登戸店	044-932-1000	●				
ブックポート203栗平店	044-988-2033	●				
平坂書房WALK店	0468-25-5537	●	●			
文教堂書店西之宮店	0463-54-2880	●				
文教堂書店伊勢原店	0463-59-5121	●				
文教堂書店平塚店	0463-31-3911	●	●			
文教堂書店秦野店	0463-84-6960		●			
島森書店大船店	0467-46-3841	●				
前田書店銀座通り店	0466-36-7602	●				
有隣堂藤沢店	0466-26-1411	●	●			
文華堂湘南台店	0466-43-0130	●				
文教堂書店六会店	0466-82-9610	●				
伊勢治書店	0465-22-1366	●	●			
川上書店ルミネ店	0467-87-3827	●				
ブックスアミ本店	0427-43-0250	●	●			
文教堂書店星ケ丘店	0427-58-6121	●				
サクラ書店秦野店	0463-82-8684	●				
伊勢原書店秦野店	0463-83-0841	●	●			
モトキ書房ソニー店	0462-29-1347	●				
石村パルコ店	0462-96-8300	●				
有隣堂厚木店	0462-23-4111	●	●			
文教堂書店中央林間店	0462-75-4165	●				
内田書店相模大野店	0462-34-8338	●				
BKオオトリ	0462-63-3856		●			
新潟県						
紀伊國屋書店新潟店	025-241-5281	●	●			
萬松堂	025-229-2221	●				
宮脇書店駅南店	025-286-5816	●				
ブックセンター長岡	0258-36-1360	●				
覚張書店	0258-32-1139	●	●			
富山県						
清明堂書店	0764-24-4166	●	●			
文苑堂書店本郷店	0764-22-0552	●				
文苑堂書店	0766-21-0333	●				
文苑堂書店横田店	0766-21-0431	●				
石川県						
うつのみや本店	0762-34-8111	●				
うつのみや御経塚店	0762-40-4067	●				
王様の本	0762-46-5325	●	●			
福井県						
勝木書店	0776-24-0428	●	●			
勝木書店二の宮店	0776-27-4678	●				
勝木書店敦賀店	0770-24-2766	●				
勝木書店鯖江店	0778-52-2828	●				
宮脇ハロー店	0776-51-8060	●				
福井大学生協書籍部	0776-21-2956	●	●			
山梨県						
朗月堂	0552-28-7356	●				
文教堂書店甲府店	0552-22-4600	●				
BOOKS KATOH	0555-23-6062	●				
長野県						

品切れの可能性もございますので，あらかじめ各書店にお問い合わせのうえ，お買い求めください

常…CQ出版社の商品(一部)常備店
D…Design WAVEシリーズ展示店
(FLEXデザインキット/CQ版，PARTHENON/CQ版)
規…最新半導体規格表展示店
P…PSpice/CQ版展示店
デ…各社半導体データブック展示店

	常	規	D	P	デ
平安堂長野店 0262-26-4545	●				
平安堂若槻店 0262-43-4545	●				
長谷川書店 0262-26-2122	●	●			
鶴林堂書店 0263-32-5340	●	●			
鶴林堂書店南店 0263-27-6000	●				
ブックスロクサン 0263-35-5555	●				
西沢書店 0268-22-0024	●	●			
ブックストア王様の本 0268-25-1235	●				
大阪屋あさま店 0267-67-5454	●	●			
平安堂飯田店 0265-24-4545	●				
笠原書店 0266-52-5443	●	●			
岐阜県					
自由書房 0582-65-4301	●	●			
大洞堂ブックス258 0584-81-2553	●				
三洋堂書店高山店 0577-33-2834	●				
三洋堂書店多治見店 0572-24-0340	●				
三洋堂書店中津川店 0573-66-7734	●				
三洋堂書店瑞浪店 0572-68-2134	●				
コスモブックセンター 0573-25-6278	●				
コスモブックセンター恵那店 0573-25-8578	●				
三洋堂書店恵那店 0573-25-8934	●				
大洞堂ルート21店 0583-70-2526	●				
三洋堂書店可児店 0574-63-2334	●				
三洋堂書店新可児店 0574-63-5834	●				
静岡県					
静岡谷島屋書店 054-254-1301	●	●			
吉見書店 0542-52-0157	●				
谷島屋連尺本店 053-453-9121	●	●			
戸田書店幸町店 053-474-4762	●	●			
静岡大学浜松 053-473-4627		●			
文泉堂書店 053-454-6358	●				
戸田書店佐鳴台店 053 448-8288	●	●			
谷島屋三方原店 053-438-2141	●				
宝塚マルサン書店 0559-63-0350	●	●			
吉野屋 0559-23-5676	●	●			
吉野屋イシバシプラザ店 0559-24-1314	●				
戸田書店本店 0543-65-2345	●	●			
多賀書店 0543-66-2502	●				
文盛堂書店 0559-75-4229	●				
岳陽堂書店 0545-61-0123	●				
戸田書店富士店 0545-51-5121	●				
マルサン書店富士店 0545-60-3358	●				
戸田書店掛川店 0537-23-3077	●				
谷島屋本沢合店 053-586-1338	●				
愛知県					
三省堂名古屋店 052-562-0077	●	●			
近鉄星野書店 052-581-4796	●	●			
名古屋丸善書店 052-261-2251	●				●
川瀬書店 052-231-4705	●				
[illegible] 052-971-1231	●				
三洋堂書店本店 052-832-8202	●	●			
三洋堂書店かの里店 052-303-7934	●				
小幡三洋堂 052-795-1128	●				
四軒屋三洋堂 052-773-7722	●				
三洋堂書店鳴海店 052-877-3388	●				
[illegible] 052-891-9861	●				
メイブリッジ 0532-55-7211		●			
精文館豊橋店 0532-54-2345	●	●			

	常	規	D	P	デ
あおい書店豊橋店 0532-64-0554	●				
シビコ正文館書店 0564-24-8173	●				
ブックス鎌倉101 0564-54-1822	●				
あおい書店上里本店 0564-222424	●				
文泉堂書店 0586-71-2181	●				
三洋堂書店一宮店 0586-77-5734	●				
三洋堂書店半田店 0569-26-3334	●				
三洋堂書店岩成店 0568-91-8834	●				
三洋堂書店豊川店 05338-3-0334	●				
杁中三洋堂刈谷店 0566-24-1134	●	●			
ブックセンター名豊 0566-21-7121	●				
三洋堂書店梅坪店 0565-35-2334	●				
精文館書店トヨタ店 0565-33-3322	●	●			
FASもとまち精文館 0565-26-6780	●				
三洋堂書店若林店 0565-53-3409	●				
栄進堂山口店 0561-85-1130	●				
あおい書店西尾店 05635-7-1151	●				
三洋堂書店小牧山店 0568-73-3462	●				
三洋堂書店東海店 052-603-8534	●				
三洋堂書店知立店 0566-82-8234	●				
桶狭間書店 0562-92-0685	●				
あおい書店長久手店 0561-63-2424	●				
三洋堂上前津店 052-251-8334	●	●	●	●	●
名古屋大学生協北店 052-781-9819	●	●			
三重県					
別所書店駅1ビル店 0592-24-1014	●				
別所書店南郊店 0592-24-1486	●				
シェトワ白揚 0593-54-0171	●				
文化センター白揚 0593-51-0711	●	●			
古川書店 0596-28-2131	●				
三洋堂書店多度店 0594-48-6134	●				
シェトワ白揚鈴鹿店 0593-82-5221	●	●			
滋賀県					
ぶっくす村岡矢倉店 0775-63-3933	●				
びわ湖タワーブックセンター 0775-73-2277	●				
三洋堂書店大津店 0775-44-1534	●				
CBワールドフジノ 0749-24-6561	●				
書店Aカデミー 0749-65-1050	●				
CBワールド守山店 0775-82-5623	●				
京都府					
オーム社河原町店 075-221-0280	●	●			
京都駸々堂京宝店 075-223-1003	●	●		●	
丸善河原町店 075-241-2161	●				
ジュンク堂京都店 075-252-0101	●	●			
アバンティブックセンター 075-671-8987	●	●	●	●	●
恵文社西大路本店 075-691-5932	●				
オーム社竹田店 075-644-2611	●	●			
リーブル京都長岡店 075-951-2228	●				
大阪府					
ユーゴー書店 06-623-2341	●	●			
旭屋書店アベノ店 06-631-6051	●	●			
紀伊國屋書店梅田店 06-372-5821	●	●			
旭屋書店本店 06-313-1191	●	●			
オーム社書店大阪店 06-345-0641	●	●			
ヒバリヤ書店ナンバ店 06-644-5407	●				
旭屋書店難波店 06-644-2551	●	●			
紀伊國屋書店心斎橋店 06-258-3771	●				
駸々堂書店心斎橋店 06-251-0881	●	●			

	常	規	D	P	デ
駸々堂Version99 06-353-4011	●				
ツタヤ書店高槻店 0726-83-4151	●				
[illegible] 0720-23-5264	●				
不二書店 0720-31-4314	●				
ヒバリヤ書店本社 06-722-1121	●				
ブックランドすばる 0720-78-2541	●				
J&Pテクノランド 06-634-1211	●	●			●
関西大生協 06-388-7641	●	●			
栗林書房 06-724-1200	●	●			
兵庫県					
ジュンク堂三宮店 078-392-1001	●	●	●	●	●
ジュンク堂ブックセンター 078-252-0777	●	●			
流水書房パティオ店 078-792-6007	●				
ブックフォーラム西神中央店 078-992-0020	●				
ジュンク堂姫路店 0792-21-3167		●			
新興書房 0792-85-3344	●				
新興書房町の坪店 0792-93-6464	●				
誠心堂書店 0792-81-2055	●				
ブックフォーラム[illegible]店 0792-95-1016	●				
ジュンク堂明石店 078-913-8118	●	●			
ハクロ書房明石店 078-911-2501	●				
三京書店伊丹店 0727-71-0002	●				
ブックフォーラム加古川店 0794-26-6704	●				
メディカ 0794-32-4115	●				
西村書店 0790-42-5008	●				
宮脇書店三原店 0799-45-0818	●				
海文堂 078-331-6501	●				
奈良県					
水嶋書房奈良店 07435-6-5355	●				
ブックス探検島 0745-22-9058		●			
和歌山県					
宮井平安堂本店 0734-31-1331	●	●			
鳥取県					
鳥取ブックセンター 0857-23-6411	●				
米子今井書店皆生通店 0859-22-2377	●				
富士書店 0857-23-7271		●			
本の学校今井ブックセンター 0859-31-5000		●			
[illegible] 0859-32-1151	●				
島根県					
松江今井書店 0852-24-2230	●				
岡山県					
紀伊國屋書店岡山店 0862-32-3411	●	●			
丸善岡山シンフォニービル店 0862-33-4640	●				
宮脇書店備前店 0869-64-4034	●				
エビスヤ書店本店 0864-22-6022		●			
広島県					
紀伊國屋書店広島店 082-225-3232	●			●	
広島積善館 0822-48-3151	●				
九嶺書房 0823-22-9090	●				
サントーク廣文館 0849-23-9434	●				
そごうB.C廣文館 0849-28-2488	●	●			
学友社西条店 0824-23-8888	●				
山口県					
文栄堂 0839-22-5611	●				
末広書店 0836-31-0086	●				
京屋書店 0836-31-2323	●				
鳳鳴館書店 0834-31-2346	●				
徳島県					

	常	規	D	P	デ
小山助学館 0886-54-2135	●				
小山助学館大学前店 0886-26-2287	●				
小山助学館徳島駅前店 0886-25-1380	●				
小山助学館55号バイパス店 0886-26-0991	●				
宮脇書店徳島本店 0886-23-6823	●				
さくら書店国府店 0886-42-8588	●				
香川県					
宮脇書店 0878-51-3733	●	●			
宮脇書店丸亀店 0877-22-5533	●				
愛媛県					
丸三書店 0899-31-8501	●				
明屋書店アカデミヤ店 0899-41-4141	●	●			
紀伊國屋書店松山店 0899-32-0005	●	●			
明屋書店伊予店 0899-82-5522	●				
明屋書店南宇和店 0895-72-6565	●				
高知県					
金高堂書店本店 0888-22-0161	●				
片桐書店高須店 0888-82-2228	●				
福岡県					
紀伊國屋書店福岡店 092-721-7755	●				
紀伊國屋書店小倉店 093-551-8801	●				
ナガリ書店 093-521-1044	●				
井筒屋ブックセンター 093-643-5530	●				
たがみ書店 0942-35-2323	●				
書肆言潮舎 0946-24-5432	●				
佐賀県					
金華堂本店 0952-22-6178	●				
長崎県					
好文堂書店 0958-23-7171	●				
熊本県					
紀伊國屋書店熊本店 096-322-5531	●	●			
まるぶん 096-356-4733	●				
大分県					
ジュンク堂大分店 0975-36-8181		●			
本町晃星堂 0975-33-0231		●			
明屋書店羽屋店 0975-45-4380	●				
リブロよしかわ 0975-56-6691	●				
宮崎県					
中央田中書店 0985-24-3511	●				
田中書店妻ケ丘本店 0986-22-4488	●	●			
鹿児島県					
ブックス鎖座 0992-22-2974	●				
春苑堂 0992-22-2131	●				
沖縄県					
球陽堂本店専門書籍部 098-863-3752	●	●			

ディスク頒布サービスのお知らせ

以下に示す各誌バックナンバのディスク頒布サービスは継続的に行っております．ご入用の方は該当雑誌で内容・価格等を確認のうえ，所定の方法でお申し込みください．なお，各申し込み書に記載の住所がスギヤマシステム宛になっている場合は，今後はＣＱ出版㈱デザインウェーブ宛に送ってください．

CQ出版㈱デザインウェーブ　〒170 東京都豊島区巣鴨1-14-2　FAX 03-5395-2127

トランジスタ技術

'92/6	PSpice用回路図エディタ
'92/8	マシン語モニタ・ディスク
'92/9	ロジック・アナライザ
'92/9	プリンタ・ポートに接続するPROMライタ
'92/10	ロジック・モニタを制作してROMエミュレータによるデバッグを理解しよう
'92/10	CD-Gの企画と画像再生の研究
'92/11	パソコンを利用した汎用半導体カーブ・トレーサの製作
'92/12	基礎からのC言語とパソコン制御
'93/7	ディジタル・テストパターン・ジェネレータの製作
'93/8	最強F1ブランプリ・ゲーム・マシンの製作
'93/10	メタ・アセンブラ
'94/2	Z80赤外線学習リモコン
'94/3	ACパワー・コントローラ
'94/4	ロジックICチェッカ
'94/5	8チャネル・ディジタル記録計の製作
'94/8	フラッシュ・メモリ・ライタの設計と製作
'94/10	マイコン・システム設計自由自在
'94/12	トランジスタ技術データベース1994年版
'94/12	オーディオ・ビジュアル・コントローラの製作

トランジスタ技術 SPECIAL

No.20	回路シミュレータMICRO-CAPIII
No.23	PALASM2.23D
No.33	バーコード読み取りシステムの製作とハンディ・スキャナによる画像読み取り装置の製作
No.35	C言語による回路シミュレータの製作
No.45	PC98シリーズのハードとソフト
No.49	Z80マイコンのすべて

インターフェース

'92/1	68K用リモート・デバッガ
'92/2	バッチファイルをEXEファイルに変換する
'92/3	DOSエクステンダ用EXPファイルをEXEファイルに
'92/3	B-tree＋
'92/4	プロトタイプ宣言生成ユーティリティ
'92/8	パイプ機能をより高速に使う
'92/9	パソコンで学ぶ割り込みプログラミング
'92/11	作りながら学ぶコンパイラの原理
'93/1	画素数変換の原理とCによる実現
'93/1	SCSIハードディスクユーティリティSUT
'93/2	Cで学ぶISAM処理の実際
'93/4	デバイス・ドライバをEXEファイル化する
'93/5	作りながら学ぶRS-232-Cインターフェース
'93/6	オブジェクト指向を実装するSCSIユーティリティ
'93/7	作りながら学ぶパソコン拡張インターフェース
'93/9	光ディスク・ディジタル・サーボ系設計の実際
'93/11	RS-232-C/68K系CPU/減色技法
'93/11	クライアントサーバ・プログラムを自作してみる
'93/12	μITRON/バージョンアップ検査/'93IF総目次
'94/1	Verilog, HDLシミュレータSILOSIII評価版
'94/1	科学計測におけるデータ処理技法
'94/1	マニトバ版VHDL
'94/2	カオス・ジェネレータ/欧州文字変換
'94/3	Windowsプログラミング/MAKE活用技法
'94/5	計測・制御とPCインターフェース
'94/6	C＋＋によるオブジェクト指向/RS-232-Cライン・モニタ
'94/7	DSOの設計・製作/MIDIファイルをRS-232-Cから出力する
'94/8	暗号化技法/MPEG-1 Video Audioボード/ポケベル呼び出しシステム
'94/8	DSP用ICEとアセンブラの製作
'94/10	Xbase言語はC言語を越えられるか
'94/11	エラー処理の考え方とプログラミング
'94/11	PARTHENON/CQ版用FLEXセルライブラリ
'94/12	実践Verilog-HDL-Lempel-Ziv圧縮チップの設計
'95/2	以降の各号

OPENDESIGN

No.1	SU.EXE
No.2	MAC
No.3	イーサネットとTCP/IP
No.8	電子メール

トラ枝コンピュータ

'91/1	以降の各号

ご案内の雑誌，書籍は全国有名書店で販売しております

●品切れのときは，書店にお申し込みいただければ取り寄せてもらえます．その際は，送料がかかりません．また，郵便未着，荷いたみなどの事故もございませんので，なるべく書店をご利用くださることをおすすめします（ただし，お手元に本が届くまで10日～25日ぐらいかかります）．

●お急ぎの場合は，ブックサービス㈱へ注文することができます．代金は商品到着時に配達料（冊数にかかわらず380円）を添えてお支払いください．1週間くらいで届きます．

ブックサービス㈱　〒113 東京都文京区本郷1-28-10 本郷TKビル1F　TEL 03-3817-0711　FAX 03-3818-5969

●その他に，小社営業部へ直接お申し込みいただくこともできます．

ご送金は，定価に送料を添えて，現金書留，郵便振替（00100-7-10665）などの安全な方法でお願いいたします．お名前，ご住所，郵便番号，電話番号（コールサインをお持ちの方はコールサインも）をハッキリと明示してください．（郵便振替の場合，お手元に本が届くまで10日ぐらいかかります．）

CQ出版㈱　営業部
〒170 東京都豊島区巣鴨1-14-2　TEL 03-5395-2141

●巣鴨の㈶無線従事者教育協会書籍販売所でも，お求めになれます．（通信販売は行っておりません）

㈶無線従事者教育協会　書籍販売所
〒170 東京都豊島区巣鴨1-12-6　TEL 03-3941-4949

●海外へ小社出版物を送る場合は，日本I.P.S.㈱へお問い合わせください．

日本I.P.S.㈱
〒102 東京都千代田区飯田橋3-11-6 TEL 03-3238-0700

平成7年7月1日(毎月1回1日発行) 第7巻 第7号 通巻62号

☎(03)5395-2141(営業部)

雑誌 06681-7

T1006681070744